SoftwareX 23 (2023) 101517

Contents lists available at ScienceDirect

SoftwareX

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

Original software publication

Diagnostic Expert Advisor: A platform for developing machine learning models on medical time-series data

Richard Polzin [a,*], Sebastian Fritsch [b,c], Konstantin Sharafutdinov [a], Gernot Marx [b], Andreas Schuppert [a]

[a] Institute for Computational Biomedicine, RWTH Aachen University, Aachen, Germany
[b] Department of Intensive Care Medicine, University Hospital RWTH Aachen, Aachen, Germany
[c] Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Jülich, Germany

ARTICLE INFO

Article history:
Received 1 June 2023
Received in revised form 30 August 2023
Accepted 31 August 2023

Keywords:
Time-series
Heterogeneous medical data
Machine learning
Python
AI model development

ABSTRACT

Setting up data structures, parallelizing code, and creating visualizations are tasks in almost any project aiming to develop healthcare AI solutions based on heterogeneous, high-dimensional data structures. While toolkits for individual parts of this workflow exist, a solution that provides integration of all steps is rarely found. We present the Diagnostic Expert Advisor, a platform for machine learning research on heterogeneous medical time-series data that aims to provide a robust environment for the rapid development of AI applications. It integrates a local web app through which whole patient cohorts, as well as the disease evolution of individual patients, can be analyzed with integrated tools for data handling, visualization, and parallelization. The platform provides sensible defaults while being flexible and extensible to fit various projects and working styles.

Code metadata

Current code version	*V1.0.0*
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00360
Permanent link to reproducible capsule	*For example: https://github.com/JRC-COMBINE/DEA/releases/tag/v1.0.0*
Legal code license	*GPL-3.0*
Code versioning system used	*git*
Software code languages, tools and services used	*python*
Compilation requirements, operating environments, and dependencies	*Bokeh==3.1.1, Flask==2.3.2, joblib==1.2.0, numpy==1.24.3, pandas==2.0.1, tqdm==4.65.0, pygwalker==0.1.8, rich==13.3.5*
If available, link to developer documentation/manual	https://diagnostic-expert-advisor.readthedocs.io/en/latest/index.html
Support email for questions	rpolzin@ukaachen.de

1. Motivation and significance

Research projects on using medical data for artificial intelligence (AI) based technologies usually encompass data extraction, processing, and analysis at some point in their lifetime. These steps exhibit a high degree of similarity throughout projects, whereas application-specific adaptions of the workflow must be implemented, requiring standardization and flexibility. For example, data pre-processing might be necessary at load time or is executed once and saved to intermediate files. Rudimentary plots might suffice, or intricate interactive visualizations might be required. As the overall workflow of data extraction, processing, and analysis remains standardized but requires manual expert interaction, it can benefit from suitable software to enable a better and faster research [1]. The Diagnostic Expert Advisor (DEA) was developed in the "Algorithmic surveillance of ICU patients with acute respiratory distress syndrome" project [2] of the SMITH consortium as a part of the German Medical Informatics Initiative. It is a platform to start research projects that provides standardization while allowing enough freedom to fit project and tooling requirements.

As a novel model development platform, the DEA's cornerstones are data management, visualization, and parallelization. The DEA offers sensible defaults, allowing for a much faster progression to model development by providing researchers with

* Corresponding author.
E-mail address: rpolzin@ukaachen.de (Richard Polzin).

https://doi.org/10.1016/j.softx.2023.101517

Fig. 1. DEA Architecture. The figure shows the two software components of the DEA. Encounter and Cohort classes are used for data management and parallelization, while the local web application explores the data and triggers computations.

a platform with all steps already set up. Establishing a structure to adhere to offers various benefits. Researchers are encouraged to work more reproducibly and organized while generating more easily shareable work. For example, visualizations developed in one project can be quickly ported to another, as the fundamental data structure is similar. At the same time, through the DEAs flexibility, it is still possible to use the required or preferred tools whether the project calls for a specific format for data storage, a preference for some visualization tool, or simply a dependency on an exact version of a machine learning library.

Another advantage of the DEA is the built-in support for parallelization. In addition to local multiprocessing, the DEA integrates directly with High-Performance Computing (HPC) software. Researchers without an extensive IT background in data handling and analysis might be hesitant to use HPC resources available. The inclusion of parallelization-specific commands and structures can quickly result in convoluted code. The DEA separates those concerns by providing integrated and unobtrusive parallelization.

Both local multiprocessing and distributed HPC calculations can be achieved using the DEA. Researchers can write calculation methods as they would without parallelization, resulting in more readable, maintainable, and more easily sharable code.

The DEA is tailored to the medical domain. While it can be extended to different areas, existing data structures assume a focus on heterogeneous time-series data and a hierarchical organization of the data based on patient cohorts. It has been used to build a prediction model for the onset of acute respiratory distress syndrome (ARDS) in the SMITH project.

Various tools and platforms for tracking machine learning experiments have emerged recently. Examples include MLFlow [3], Neptune [4], Weights and Biases [5], and TensorBoard [6]. These programs are often focused on the tracking of experiments and parameters for the development of AI models. In contrast, the DEA is built to provide a platform for the fast integration and development of such models into the clinical context, filling a gap in the current research software landscape.

A recent article described research software engineering as a pivotal and often undervalued research area and emphasized a need for infrastructure solutions allowing data to be made "interoperable, visualized and leveraged by experts and non-experts alike" [7]. Our proposed research platform tackles these challenges at a small scope, enabling faster data comprehension and development of medical prediction models while still being accessible to medical experts and researchers without requiring an extensive background in software engineering.

The platform is built with Flask [8] at its core, providing a local web app. This web app can be used as an interface for explorative analysis or to run code in parallel and on HPC hardware. Pandas DataFrames [9] are used for internal data storage. SLURM [10] is currently supported for interacting with the HPC, and the Joblib [10] library is used for intermediate data formats. Interactive visualizations are available through Bokeh [11]. PyGWalker [12] has been integrated as a Tableau [13] like data exploration tool.

Accommodating different choices for storage patterns or libraries has been a priority during the development of the Software. Therefore, all libraries, except for Flask, can be exchanged and adjusted to fit the needs of different researchers and projects.

2. Software description

The DEA is written in Python [14]. In the standard implementation, data is saved to comma-separated value files (CSV), allowing easy interaction with other tools and languages. Such files can be opened by spreadsheet software like Microsoft Excel [15] or read in different environments like Matlab [16] or R [17]. The DEA is interfaced through a browser as a web application and the data management Python classes.

2.1. Software architecture

As shown in Fig. 1, the DEA consists of two components. A set of classes encapsulate the data to allow for parallelization, and a Flask server provides the web interface. The data management is based on wrappers around Pandas DataFrames. Thus, switching to different environments, such as Jupyter Lab [18], is possible without changing formats.

The local web application is built on Flask, a well-established micro framework for building web applications. It can be customized in various ways. Some examples are explained in the next chapter.

2.2. Software functionalities

The *encounter* and *cohort* classes allow for extensible visualization, data storage, and parallelization. An encounter describes an individual data point (in the context of our project: a patient's visit to the ICU). A cohort is a container for multiple of those encounters. They further allow for direct access to the underlying DataFrames, which can be utilized directly, e.g., for model development. Training of such models could then be run through the DEA directly on HPC infrastructure or entirely external through other frameworks using the standardized pandas DataFrames.

Using the DEA requires transforming the data to this Encounter/Cohort format. Existing data needs to be imported into Pandas DataFrames. Pandas already provides many methods to read from CSV, JSON, Excel, or Apache arrow [19] files. The quickstart guide[1] provides a sample implementation and explains these steps in detail.

[1] https://diagnostic-expert-advisor.readthedocs.io/en/latest/usage/quickstart.html

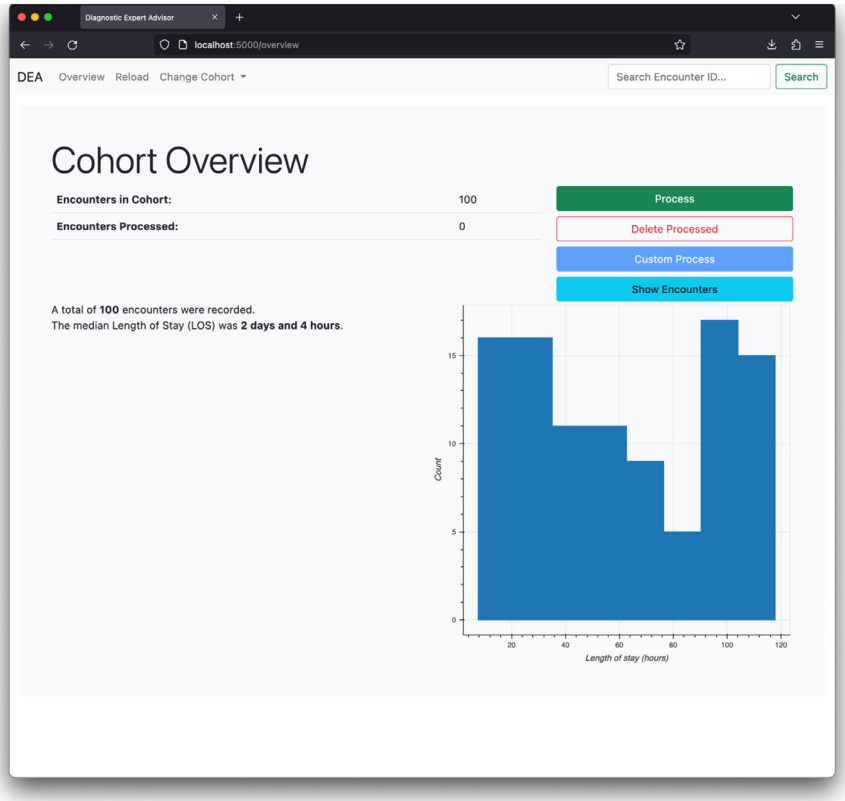

Fig. 2. Cohort Overview, showing various customized elements. This figure illustrates, on the one hand, the customizability, as the LOS calculation, encounter stats, and the corresponding plot are all such customizations. On the other hand, the "Custom Process" button shows the capabilities to add custom calculations and processes to be run on HPC.

The functionalities outlined in this chapter are further described in the developer documentation.[2] All of them are designed to be customized and adjusted to the specific research task and data.

2.2.1. Visualization

Visualizations can be created in many ways. Due to the flexibility of providing a web server, almost all of them can be embedded into the DEA. Matplotlib [18] plots can be saved to static images and shown in plain HTML, while interactive plots, such as Bokeh visualizations, can utilize custom JavaScript. Plots can be defined on encounters and cohorts, providing different levels of detail. Cohort visualizations could include general population statistics, while encounter visualizations could focus on disease-relevant parameters and individual events in the ICU.

2.2.2. Explorative analysis

Through the visualizations, it is already possible to explore individual encounters in detail with the DEA. To further this capability and provide an effortless way to prototype new visualizations quickly, we integrated PyGWalker in the DEA for a tableau-like explorative data analysis environment. Through this component, researchers can interactively create visualizations on the fly to test hypotheses or develop new ones. Plotting parameters without coding further encourages medical professionals and interdisciplinary researchers to investigate data.

2.2.3. Filtering

The DEA can load distinct cohorts, switching between ICU wards or hospitals. It is further possible to filter the encounters dynamically. Through filters, it is possible to explore different sub-cohorts of encounters and examine, e.g., especially endangered patients. Filters could also be used to evaluate model performance on different subsets of data. For example, in our project, we split the cohort into patients with varying lung failure levels to evaluate ARDS prediction models.

2.2.4. Parallelization

While many operations would benefit from parallelization, the increased effort during development often hinders its actual implementations. To encourage parallelization, the DEA provides a way to run calculations in parallel. These can be run either locally or on HPC hardware with limited setup.

3. Illustrative examples

Upon starting the DEA, the user is shown a screen where a cohort can be selected. Afterward, an overview page is displayed. Fig. 2 shows the various custom information for the cohort on this screen. It is possible to start calculations on the whole cohort through the "process" and "custom process" buttons. Both buttons trigger specific calculations on the HPC Cluster. The length-of-stay plot and all other information, like "Encounters in Cohort" and "Encounters Processed", can be fully customized. Badges can be added to provide a quick visual indication, e.g., that all data has been processed.

To inspect individual encounters, they can be searched through the navigation bar at the top, or an overview can be shown by clicking "Show Encounters". Fig. 3 shows the corresponding UI. Custom information tags can be displayed per encounter to provide a quick overview. This allows, for example, encounters from extremely sick patients or those with especially

[2] https://diagnostic-expert-advisor.readthedocs.io/

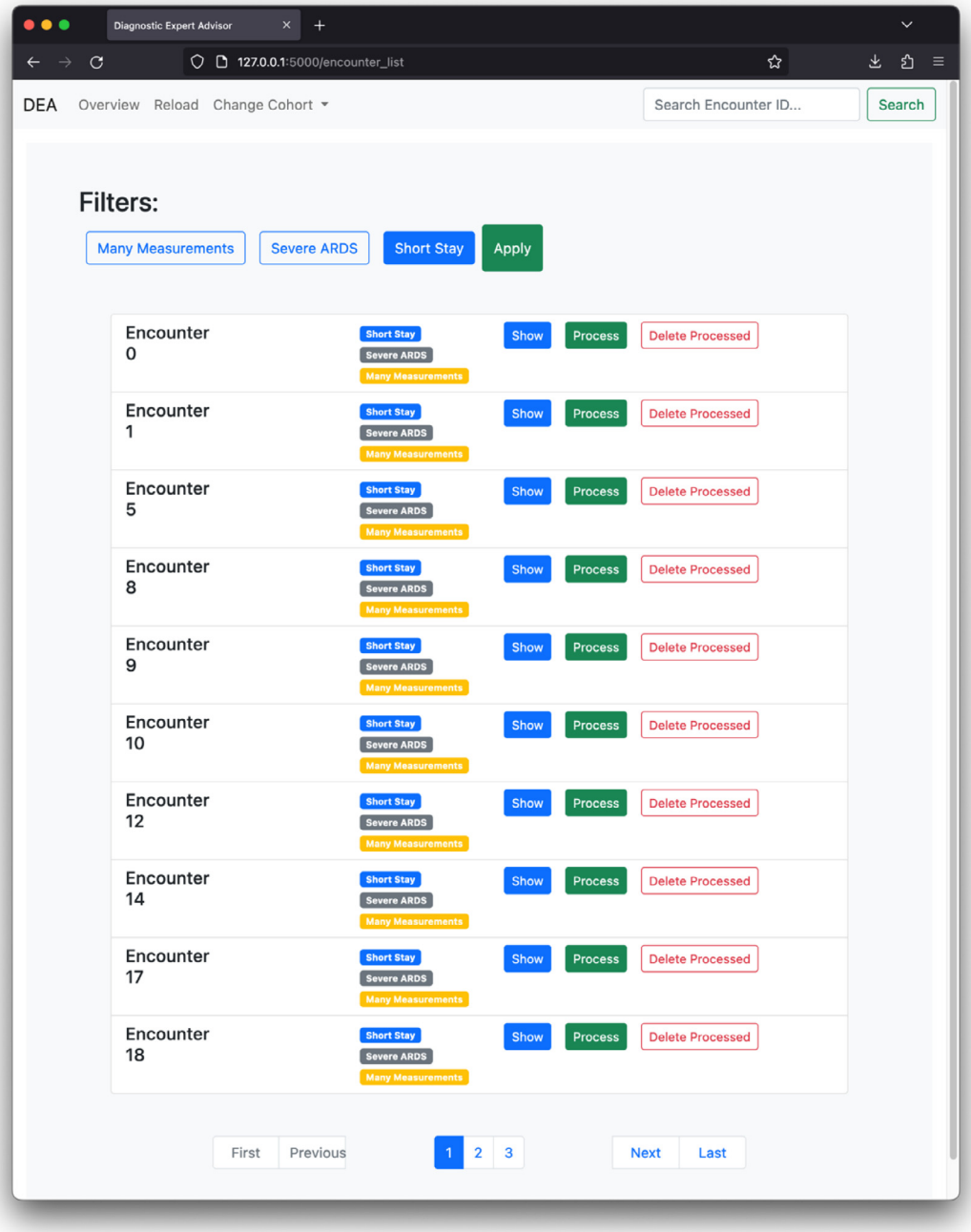

Fig. 3. Encounter Overview showing matches for the active filter "short stay". Filters, as well as tags, are fully customizable. This view presents the primary way to browse patient data in the DEA, allowing for fast visual identification of relevant patients through tagging and the execution of various pertinent commands during research and model development, such as re-running processing or removing intermediate files.

bad prediction results to be more visible. Filters for the selection of sub-cohorts are also available on this page.

Calculations on individual encounters can be re-run, and all visualizations, including the PyGWalker Interface, are available when selecting a particular encounter, as seen in Fig. 4.

4. Impact

The DEA provides a free and open-source research platform for developing AI models.

We hope the DEA can help improve the quality and speed of research work by promoting open collaboration, streamlining project setups, and enabling more straightforward HPC utilization. Lowering the required expertise in software engineering allows medical professionals focusing on research to utilize such resources more quickly.

Currently, the researchers involved in its development and adjacent groups have chiefly used the DEA. We plan to further evaluate its impact on operational practices and outcomes in the context of clinical studies in the future. Its design has been steered by close cooperation with medical experts in the scope of the SMITH project [20,21]. The platform will, for example, be set up as a demonstrator for the National High-Performance Computing for Computational Engineering Sciences (NHR4CES) Simulation and Data Lab, Digital Patient,[3] to showcase the ease

[3] https://www.nhr4ces.de/simulation-and-data-labs/sdl-digital-patient/

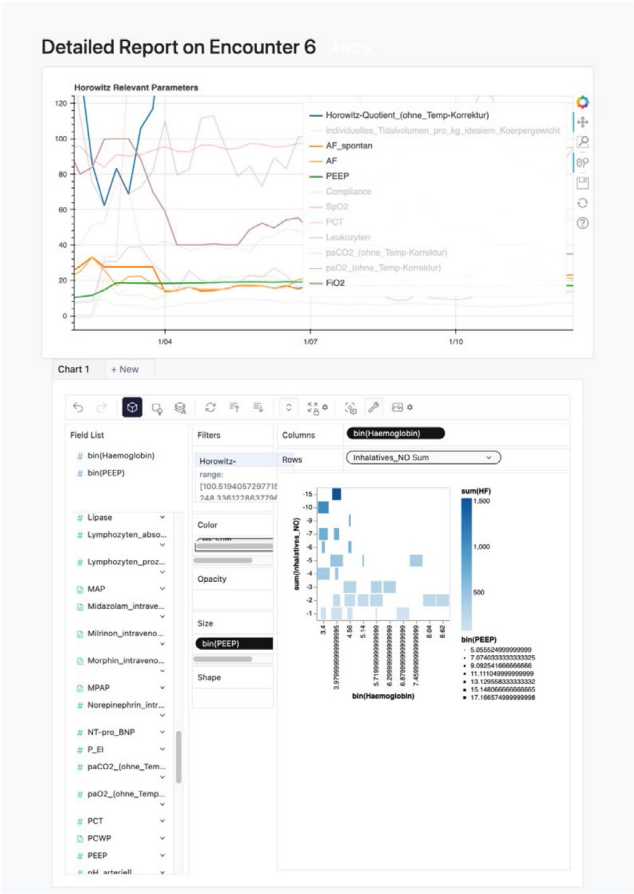

Fig. 4. Default bokeh plot with ARDS-specific parameters shown, as well as the PyGWalker interface. This figure shows detailed information about a specific patient encounter in the ICU. It offers interactive plots to explore the data intuitively. The PyGWalker interface enables users to generate custom plots on the fly without touching code. This allows for a very rapid pace of data exploration and individual parameters to be explored and plotted against one another as needed.

of utilizing HPC resources to medical researchers. It will be facilitated and extended as a model development platform for the EDITH European Virtual Human Twin project.[4]

While the DEA is designed around heterogeneous medical time-series data, we hope its modularity and adaptability encourage other researchers to fork the project and create adaptations for their respective domains, spreading the concept further.

5. Conclusion

The Diagnostic Expert Advisor (DEA) provides a platform for developing machine learning models on heterogeneous medical time-series data. It enables researchers to use a streamlined workflow and encourages a more organized setup. Making parallelization and interactive visualizations more approachable supports the more widespread adoption of these features. The innate data structure and the ability to quickly investigate individual encounters foster patient-centric research. Our platform provides a foundation for many medical research areas while staying generic enough to be adaptable to other domains.

The platform is designed to be flexible and adaptable. Whether data is extracted directly from a hospital database or cross-referenced from another project, both approaches can be accommodated with few changes to the DEA.

Our broader vision extends to an adaption of similar platforms to different domains. Establishing such platforms results in more organized, readable, and shareable research, benefitting the whole community.

Funding

This publication of the SMITH consortium was supported by the German Federal Ministry of Education and Research, Germany (Grant Nos. 01ZZ1803B and 01ZZ1803M).

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

No data was used for the research described in the article.

References

[1] Anzt H, Bach F, Druskat S, Löffler F, Loewe A, Renard B, et al. An environment for sustainable research software in Germany and beyond: current state, open challenges, and call for action [version 2; peer review: 2 approved]. F1000Research 2021;9. http://dx.doi.org/10.12688/f1000research.23224.2.

[2] Marx G, Bickenbach J, Fritsch SJ, Kunze JB, Maassen O, Deffge S, et al. Algorithmic surveillance of ICU patients with acute respiratory distress syndrome (ASIC): protocol for a multicentre stepped-wedge cluster randomised quality improvement strategy. BMJ Open 2021;11:e045589. http://dx.doi.org/10.1136/bmjopen-2020-045589.

[3] Zaharia M, Chen A, Davidson A, Ghodsi A, Hong SA, Konwinski A, et al. Accelerating the machine learning lifecycle with mlflow. 2018.

[4] Neptune ai. Neptune.ai : experiment tracking and model registry. 2022, https://neptune.ai.

[5] Biewald L. Experiment tracking with weights and biases. 2020, https://www.wandb.com/.

[6] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. Tensorflow: large-scale machine learning on heterogeneous distributed systems. 2016, http://dx.doi.org/10.48550/arXiv.1603.04467.

[7] Horsfall D, Cool J, Hettrick S, Pisco AO, Hong NC, Haniffa M. Research software engineering accelerates the translation of biomedical research for health. Nat Med 2023;29:1313–6. http://dx.doi.org/10.1038/s41591-023-02353-0.

[8] Flask. Python. 2023, https://github.com/pallets/flask.

[9] pandas-dev/pandas. Pandas. 2020, http://dx.doi.org/10.5281/zenodo.3509134.

[10] Joblib Development Team. Joblib: running python functions as pipeline jobs. 2020, https://joblib.readthedocs.io/.

[11] Bokeh Development Team. Bokeh: python library for interactive visualization. 2018.

[12] PyGWalker. Python. 2023, https://github.com/Kanaries/pygwalker.

[13] Tableau: Business intelligence and analytics software. Tableau; 2023, https://www.tableau.com/node/62770 (accessed April 28, 2023).

[14] Van Rossum G, Drake Jr. FL. Python reference manual. Amsterdam: Centrum voor Wiskunde en Informatica; 1995.

[15] Microsoft Corporation. Microsoft excel. 2018, https://office.microsoft.com/excel.

[16] Toolbox SM, et al. Matlab. Mathworks Inc; 1993.

[17] R Core Team. R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing; 2022.

[18] Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng 2007;9:90–5. http://dx.doi.org/10.1109/MCSE.2007.55.

[19] pmc. Apache arrow 9.0.0 release. Apache Arrow; 2022, https://arrow.apache.org/blog/2022/08/16/9.0.0-release/ (accessed April 28, 2023).

[20] Fritsch S, Maassen O, Riedel M. [Artificial intelligence: Infrastructures and prerequisites at European level]. Anasthesiologie Intensivmed Notfallmedizin Schmerzther AINS 2022;57:172–84. http://dx.doi.org/10.1055/a-1423-8052.

[21] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, et al. Future medical artificial intelligence application requirements and expectations of physicians in german university hospitals: Web-based survey. J Med Internet Res 2021;23:e26646. http://dx.doi.org/10.2196/26646.

[4] https://www.edith-csa.eu/