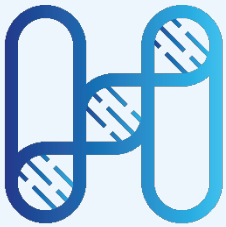


HEREDITARY

HetERogeneous sEmantic Data integratIon for the guT-bRain interplaY

Deliverable 5.1

Visualization components for sequences, networks, text, and high-dimensional data

Version 1.00, 16.12.2024

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No GA 101137074. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Funded by the European Union

EXECUTIVE SUMMARY

Research in HetERogeneous sEmantic Data integratIon for the guT-bRain interplaY (HEREDITARY) involves understanding properties, relationships and patterns in large and complex, heterogeneous data, to obtain knowledge from biomedical data sets. Interactive data visualization techniques help to overview and make sense of large and complex data sets, by finding appropriate visual representations and user interactions for the data. Also, interaction allows to drill down and view the data from different perspectives. Importantly, we can include data analysis algorithms into visualizations which can then be interactively controlled by the analyst. Resulting are Visual Analytics workflows which support domain users to effectively and efficiently analyze data.

Work Package 5 in HEREDITARY focuses on the research and development of effective tools for interactive visual data analysis, supporting the HEREDITARY use cases. This deliverable presents the results achieved in the first 12 months of the project, in developing visualization components for non-spatial data. The components have been informed by data and use cases from HEREDITARY partners and available open research data. The components are the basis for building standalone applications to address specific HEREDITARY use cases in the next project steps, including functionality for interactive search and user guidance.

DOCUMENT INFORMATION

Deliverable ID	**5.1**
Deliverable Title	**Visualization components for sequences, networks, text, and high- dimensional data**
Work Package	**WP 5**
Lead Partner	**TU Graz**
Due Date	**31.12.2024**
Date of submission	**16.12.2024**
Deliverable Type	**R – Report**
Dissemination level	**PU – Public**

AUTHORS

Name	Organization
Tobias Schreck	TUGRAZ
Stefan Lengauer	TUGRAZ
Peter Waldert	TUGRAZ
Benedikt Kantz	TUGRAZ
Michael Grabner	TUGRAZ
Lukas Schilcher	TUGRAZ
Dorian Percic	TUGRAZ
Casper van Leeuwen	SURF
Matteo Lissandrini (Reviewer)	AAU
Anna Romanovych (Contributor)	UNIPD

REVISION HISTORY

Version	Date	Author	Description
0.10	2024-06-04	TUGRAZ	Structure for deliverable
0.20	2024-11-25	All authors	Complete draft including all components
0.30	2024-11-30	AAU	Reviewer has provided a list of suggested improvements to the deliverable
0.40	2024-12-09	All authors	Resolved reviewers comments
1.00	2024-12-16	UNIPD, TUGRAZ	Resolved Review the final formatting of the document

Contents

List of Figures

Acronyms

ALS Amyotrophic Lateral Sclerosis. 6, 7, 13, 19, 23, 32

BRAINTEASER Bringing Artificial Intelligence home for a better care of amyotrophic lateral sclerosis and multiple sclerosis. 32

ETL Extraction, Transform and Load. 34

GMM Gaussian Mixture Models. 28

HEREDITARY HetERogeneous sEmantic Data integratIon for the guT-bRain interplaY. 3, 7–10, 12, 17, 29, 30, 32, 33, 35–37

HERO HEREDITARY Ontology. 7, 32, 34, 35

HMM Hidden Markov Model. 31

iDPP Intelligent Disease Progression Prediction. 6, 13, 19

LLM Large Language Model. 24

MS Multiple Sclerosis. 6, 7, 13, 23, 32

OnSET Ontology and Sparse data Exploration Tool. 6, 7, 13, 32–35

PCA Principal Component Analysis. 22, 24

PD Parkinson's Disease. 6, 7, 13, 23, 24

PGO Performance-guided optimisation. 30

UMAP Uniform Manifold Approximation and Projection. 22

WASM WebAssembly. 30

1 Introduction

We introduce interactive visualization for data analysis, surveying key approaches and methodologies for application-oriented visualization and visual analytics designs. Important visualization components have been developed as the basis for application building for the Visual Analytics platform in HEREDITARY. The components are described in the following in general, and in more detail in Section 2.

1.1 Visualization and Visual Analytics: Value and Methodologies

Visualization and the field of Visual Analytics have an important role in understanding large and complex data sets, in many different applications. Visual Analytics techniques support data analysis by closely integrating the visualization of data and models with machine learning methods in an interactive way [Keim et al., 2010; Sacha et al., 2014; Thomas and Cook, 2005]. Visual Analytics supports tasks such as data exploration, hypothesis generation, visual communication of analysis results, and the selection of appropriate training data and parameters for analysts. Properly designed, visualization can add value to the data analysis process in many ways [Fekete et al., 2008; van Wijk, 2005]. It includes leading to novel insights, supporting data understanding, inspiring searchers, and also to communicate found patterns and insights to others.

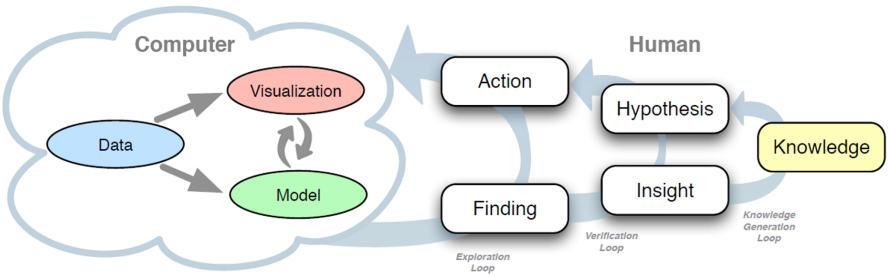


Figure 1: The visual analytics process suggests to go from data to knowledge, using interactive data visualization, tightly integrated with data analysis and modeling approaches. Appropriately designed visual analytics applications support the user in achieving findings and insights in their data, which can lead to new hypotheses being generated. Eventually, actionable knowledge can arise. Figure taken from [Sacha et al., 2014].

Visualization and Visual Analytics are established fields today, with rich experience how visualization, interaction and visual analytics workflows can be implemented [Munzner, 2014; Tominski and Schumann, 2020; Ward et al., 2010]. The field has developed methodologies to create effective visual analysis applications for different domains. The *Design Study* methodology [Sedlmair et al., 2012] suggests to work closely with domain stakeholders and experts, going through iterative cycles of learning and understanding their tasks, data used, and methodology; and iteratively design and introduce interactive visualization where it is useful to support the domain work. The visualization designs can be drawn from existing visualization techniques, which can also be adapted. Additionally, depending on the case and application, novel visualization techniques need to be developed, suited for the tasks and data at hand. The *Design Triangle* methodology [Miksch and Aigner, 2014] is a valuable tool to obtain requirements and to inform the visualization design. It suggests to characterize the data, the users, and the tasks at the beginning of the project, and iteratively develop, evaluate and improve the designs. Throughout the development process of all the components in this deliverable, we were in continuous contact with project partners and we informally obtained first requirements with respect to data,

users, and tasks, which informed the components designs. The components are the basis for the continuous requirement analysis and specification taking place in Task 5.5 (Requirements analysis, evaluation, and user studies) from the second project year on.

Visual Analytics methods are widely adopted in many application areas. Visualization to date has been successfully applied in many biomedical problems, as recent surveys show, e.g., in [Gillmann et al., 2021; Kozlíková et al., 2017; Kreiser et al., 2018; Lawonn et al., 2018; Oeltze-Jafra et al., 2019; Pretorius et al., 2017]. Respective techniques are regularly published in the key visualization conferences, e.g., IEEE Visualization and Visual Analytics (VIS), Eurographics Conference on Visualization (EuroVis), IEEE Pacific Visualization Conference (PacificVis), etc., and journals, e.g., IEEE Transactions on Visualization and Computer Graphics, Computer Graphics Forum, etc. Topical venues exist, e.g., the Workshop Series on Visual Analytics in Healthcare, and the Eurographics Workshop Series on Visual Computing for Biomedicine, among others.

In WP5, we build on established knowledge and our own research efforts to implement a number of software components for visualization of particular data of relevance to the HEREDITARY use cases. These are the subject of this report. In the following, we will develop visualization applications using and extending these components, to support research regarding the use cases. Eventually, the set of tools will form a visualization platform for the heterogeneous HEREDITARY data sets.

1.2 Components

HEREDITARY WP5 is concerned with developing interactive visualization techniques for supporting domain researchers working on the different use cases in HEREDITARY. We follow the above established methods to develop, evaluate and improve visualization applications. In this deliverable, we describe the results of developing visualization components wich form building blocks to stand alone visualization applications, which will be composed in the next project phases. We follow the design triangle in making the design choices. We note data and uses cases / tasks are being established during the project runtime. We focused on HEREDITARY data where possible, and focus on existing open source and challenge data for other cases which are being developed at present.

Time-dependent and sequence data. Sequences of values occur in many analysis contexts, describing e.g., measurements over time, e.g., neuron activity levels over time, or along a spatially defined sequence, e.g., bases in a DNA sequence. To date many visualization techniques for this data type have been proposed [Aigner et al., 2023]. Our components support line chart visualization of neuron activity levels (Section 2.1) and multivariate measurements of cell activity (Section 2.6).

Network data. Networks and relationships occur in many data sets, and often give rise to complex structures. Established visualization techniques for networks typically apply node-link based designs, and matrix-based designs [von Landesberger et al., 2010]. Network structure is also subject to change over time, for which specialized visualizations can be applied [Beck et al., 2017; Cakmak et al., 2021]. Our components support visualizing brain connectome networks (Section 2.1) and ontology data using node-link and circular TreeMap visualization (Section 2.7).

Image data. Imaging techniques produce rich image data from specimens, e.g., using X-Ray, CT and MRI techniques. This data can also be refered to as spatial data, as measurement values are located at discrete locations in a 2D or 3D measurement space (giving rise to images and volumes). Visualization of images and 3D voluments is established in many data visualization tools, including e.g., Visualization Toolkit (`https://vtk.org/`). Values to show per location can include scalar values, e.g., intensities, or multidimensional

values, e.g., RGB color information, directional/vector information, and other measurements depending on the imaging technology. In effect we need to combine image visualization with further visualization techniques, such as high-dimensional visualization techniques to fully explore the data. Our components support the visualization of 2D images for brain activity data linked to distribution data (Section 2.2), and overlaid high-dimensional data (see the Droplet technique in deliverable 5.3).

High-dimensional data. Oftentimes, measurements and data in biomedical data analysis comprise high-dimensional data, i.e., where more than one value are given for the observation. Patterns can be found in any of the dimensions, or in combinations of dimensions. The analysis problem is challenging as the number of subspaces in which patterns could exist grows exponentially with the dimensionality, sometimes called the curse of dimensionality. Visualization to date has developed many techniques for visualizing high-dimensional data [Munzner, 2014; Tominski and Schumann, 2020; Ward et al., 2010]. Examples are parallel coordinate plots, scatter plot matrices, recursive patterns, etc. Also, using visual markers such as color, shape, texture, or abstract representations (glyphs) to encode additional dimensions besides location in 2D or 3D plot, is possible. Dimensionality reduction can be applied to reduce the data to a smaller set of dimensions, which can capture correlations and exclude irrelevant dimensions from the analysis.

Several of our components support high-dimensional data: dimensionality reduction of high-dimensional microbiota distribution data (Section 2.2) and high-dimensional cell type distribution data in image data (see the Droplet technique in deliverable 5.3). In addition, a series of visualizations using bar charts, tables, scatter plots, and heatmaps support analysis of high-dimensional empiric data in the ALS, PD and MS example data sets (Sections 2.3, 2.4, and 2.5).

Textual data. Text is very important as it encodes information ranging from item labels in a diagram, to symbols forming sequences, up to document collections. Many visualization techniques have addressed visualization of text features, named entities, sentence relationships, sentiments expressed, clusters of documents etc. An overview of different techniques can be found in [Kucher and Kerren, 2019] and the text visualization browser (`https://textvis.lnu.se/`). Our components currently support text visualization as data labels (Section 2.2 and other components) and in particular Ontology visualization networks showing relationships among concepts and entities, extracted from documents and by data analysis (Section 2.7).

Spatial, image, and simulation data. Heterogeneous biomedical data also includes spatial and image data, as well as complex data arising from simulation. Please refer to the corresponding deliverable D5.3 for respective components.

In the following components, we indicate the type of data supported by abbreviations NETWORK, TIME, IMAGE, HIGH-DIM, TEXT in the headings.

1.3 Structure of the component descriptions

The components in this deliverable are described in Section 2. Note specific data types are supported by more than one component. Also, several components support two or more data types at the same time. This is owing to the heterogeneous nature of many biomedical data sets and analysis tasks. In each case, the components are representative for specific types of analysis, e.g., understanding both properties in an image space, and a corresponding high-dimensional space, and understanding how both spaces interact. For example, please see Section 2.2.
The components are described using the following structure:

1. **Considered Data / Test Dataset / Task Description:** For each component, we base the design on data and analysis tasks from HEREDITARY, or where not yet available, on related tasks and data from open sources. The task description is an indicative example, and to be refined in the future work where applications will be built from the components.
2. **Visual Analytics Approach:** Description of the used visualization types, and how users can interact with the visualization, forming a visual analytics process (see also Section 1.1).
3. **Implementation Description (Backend, Frontend, Development Environment).** This describes implementation details. We mainly follow a web-based development approach, where the main data processing and algorithmic analysis is done in a backend, and a web-based frontend supports the visualization and interaction. We use typical development environments from web stack development, including Python, Node, HTML/CSS, D3, etc.
4. **Demonstration of Preliminary Use Case(s).** We exemplify the component application on test data, and outline potential usages of the components based on HEREDITARY or open data sets.
5. **Development Roadmap: What to Develop Next?** The components are the basis for combining and extending them to form standalone applications supporting HEREDITARY use cases. We describe the next steps. Generally, the applications will be extended by interactive search functionality, user guide approaches, and be evaluated and improved.

2 Visualisation Components

The following visualisation components have been developed:

Components

2.1 Brain Connectome and Activity Data: Brain Simulation Data Set from the 2023 IEEE SciVis Challenge (NETWORK, TIME)

This component is concerned with the IEEE SciVis Contest 2023 - Dataset of Neuronal Network Simulations of the Human Brain [Gerrits et al., 2024].

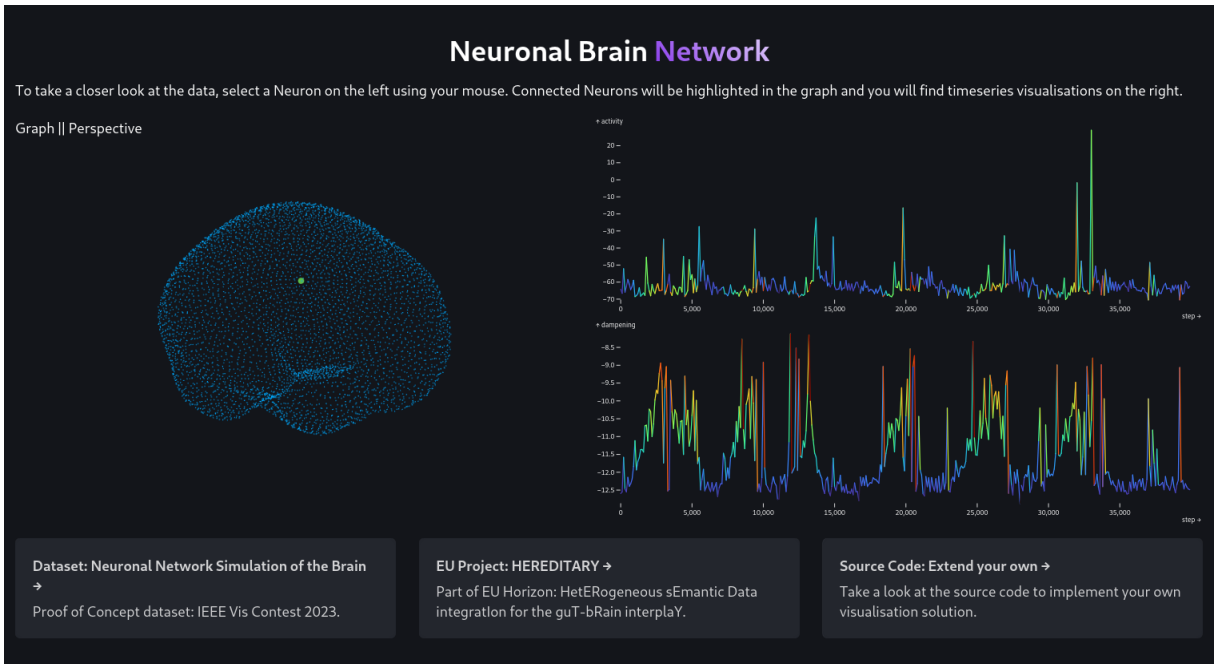


Figure 2: Visualisation interface between the 3D-embedded neuronal connectome of the brain and the time-series of its activity on the right. Neurons can be selected simply on-click from the graph / perspective view on the left, following a detail-on-demand approach. Using the mouse, one can navigate the currently active perspective view through panning, utilizing scroll for zooming.

2.1.1 Considered Data / Test Dataset / Task Description

The dataset contains a detailed simulation of 50,000 neurons in a human brain. The simulation methodology is described in [Rinke et al., 2018]. For example, over time $t \in \mathbb{R}^+$, each neuron's calcium level $\mathrm{Ca}(t) \in \mathbb{R}$ is modelled according to the following ordinary differential equation,

$$\frac{\mathrm{dCa}}{\mathrm{d}t} = \begin{cases} -\frac{\mathrm{Ca}(t)}{\tau} + \beta & \text{if neuron fires} \\ -\frac{\mathrm{Ca}(t)}{\tau} & \text{else} \end{cases}$$

with τ the calcium decay constant and β the calcium intake constant. Similar equations describe the remaing quantities in question, cf. [Rinke et al., 2018] and the challenge description [Gerrits et al., 2024]. The full recording of each neuron over time contains the following details: step, fired, fired fraction, activity, dampening, current calcium, target calcium, synaptic input, background input, grown axons, connected axons, grown dendrites and connected dendrites. Each neuron has weighted input and output pathways to other neurons, transmitting signals. Input connections can receive signals, output connections can transmit a neuron's firing activity on to further neurons. These connections can change over time, which is another challenge for the data representation, yet the visualisation. Next to their network connections, neurons are also modelled with

concrete positions in space which makes it a unique approach to such visualisations. The goal is to enable a whole view on the higly complex dataset in question, balancing a fluent, responsive overview with guided explorability.

2.1.2 Visual Analytics Approach

The view (Figure 2) is split into two halves: On the left, we have a shared view between the neuron graph and a perspective view. The changing connectivity of the neuron network motivates a graph view, while the 3-dimensional embedding of the neurons gives rise to a navigatable perspective view on the dataset. One can switch between the two using the buttons provided above the view (*Graph* || *Perspective*). On the right, the varying quantities described above are displayed, for one selected neuron, as a coloured line chart. The colour intensity is used to indicate areas of high *activity*, although that can be adjusted.

The graph view's layout is based on a projection of the 3-dimensional positions of the neurons down to two selectable axes. Neurons can then be selected on-click with a snapping tooltip that makes the selection process smoother.

2.1.3 Implementation Description (Backend, Frontend, Development Environment)

The implementation uses the Astro web framework, based on vanilla JavaScript (ECMAScript 6 standard specification, which many of the following components are also based on) and runs in the browser. Pre-processed data is then fetched on-demand to keep memory usage low at all times during the process. Specifically, that is the connectivity table along with positions, and the selected neuron's timeseries data. The graph view is an extension of ObservableHQ's *Plot* library, while the perspective view is implemented on top of *three.js* using an automatically constructed mesh based on the pre-fetched positional data. The timeseries including the coloured indiation, are again based on the *Plot* library. Internal processing of the data is handled on the frontend using the *d3* library.

2.1.4 Demonstration of Preliminary Use Case(s)

The visualisation provides a great overview of the data at a glance, on a single screen. Using the perspective view, the user gets a good impression of the structure of the brain, and it is straightforward to take a closer look at individual sections and clusters within the dataset. Due to the technical setup of the simulation, many neurons are clustered together, which from afar appear as single entities. Upon zooming in, the visualisation reveals structure within the dataset that was not apparent before. The colouring of the timeseries view then facilitates analysis of the temporal nature of the neurons, always comparing two different quantities over time allowing the user to gain a concrete understanding of the dataset and simulation. For example, red areas in the timeseries charts indicate high dampening coefficients which can be put in direct relation to the activity of the individual, currently selected neuron of the brain, within one view.

2.1.5 Development Roadmap: What to Develop Next?

Selection and highlighting features in the perspective view still needs improvement, which is difficult due to the high number of neurons and their spatial cluttering. Neurons, at the moment, do not have any volume so that when zooming in, one can see details of close-by neuron clusters. This makes selection of the neurons through raytracing much harder. Further goals include the finalisation of a lensing feature in the graph, as well as the perspective view, which aims to tackle this problem as a by-product of a richer spatial visualisation. The interface's responsiveness could also be improved utilising the in-browser cache.

2.2 Gut microbiota (HIGH-DIM)

This component is motivated by the works of Nils Kohn [Kohn et al., 2021] (WP2/4, RUMC) on the relation of gut microbiota and large-scale brain network connectivity, i.e., the gut-brain axis. In an integrated visualization (Fig. 3), they try to convey how gut microbiota and brain networks covary together. There are established visualization techniques for illustrating both the contributions of different modalities and brain activity in the form of treemaps and characteristic brain slices (left and middle, respectively). For the gut-microbiota (right), however, there is to our knowledge no such standard visualization. In the given example, the most relevant biota, according to a chosen cutoff, are shown as an enumeration with the sign of their relation – positive or negative – indicated by an optional "(-)" at the end.

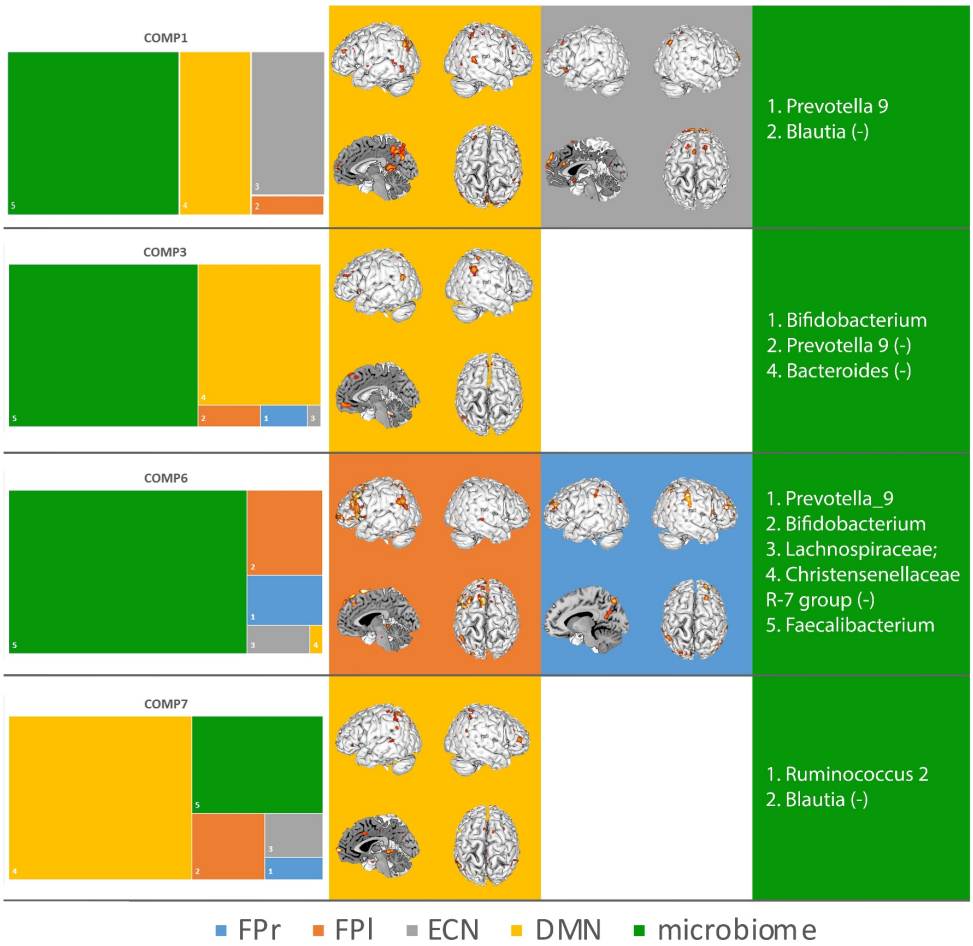


Figure 3: The integrated visualization of contribution of different modalities with a proportional display in the form of treemaps (left), brain slices through the MRI images (middle) and enumerations for the gut microbiota (right). *Image taken from Kohn et al. [Kohn et al., 2021].*

With our proposed design, we want to additionally show quantities of the respective contributions, their signs, and the relations of gut microbiota among themselves.

2.2.1 Considered Data / Test Dataset / Task Description

The data used in our experiments was provided by HEREDITARY partner RUMC (Nils Kohn). In total, the provided data comprises of 25 components describing 3 modalities. The 189 considered biomes, such as *Prevotella 9*, *Blautia*, *Bifidobacterium*, etc., are described by 143 features. For each biome we have the information how it contributes to each of the components.

The given dataset also contains 4D brain networks, with the 4th dimension being the modality. However, this information was not used in our experiments.

2.2.2 Visual Analytics Approach

Our idea is to apply dimensionality reduction. Specifically, to use the microbiota's feature vectors to project them into 2D space, such that their proximities reflect their respective similarities. At the same time, we use the respective loadings to show on a per-component basis their contributions. To this end, we use color coding and scaling.

2.2.3 Implementation Description (Backend, Frontend, Development Environment)

The visualization prototype is a standalone Python implementation, using the Matplotlib[1] visualization library. For data (pre)processing and linear algebra operations, the *NumPy*[2] package for numerical calculations was employed.

2.2.4 Demonstration of Preliminary Use Case(s)

Fig. 4 shows a result of applying our implementation on the provided data. Each component features the same positioning of biota, but their contributions are illustrated by both color and scale. It can be easily seen that a few components, i.e., Comp. 7, 11, 17, the gut microbiota is particularly relevant. Also it can be observed that in certain components (7 and 17) distinct similarity groups of microbiota covary together, while random patterns can be observed in other cases (Comp. 11).

2.2.5 Development Roadmap: What to Develop Next?

The current prototype implementation outputs a static visualization. Our next steps include the introduction of interactivity in the form of filtering, detail-on-demand systems, sorting, etc.

Also, a major milestone will be to merge it with the work of HEREDITARY partner SURF (Casper van Leeuwen), who have developed a complementary type of visualization (matrix visualization of the microbiota) with rich interaction support (see Deliverable 5.3).

[1] https://matplotlib.org/
[2] https://numpy.org/

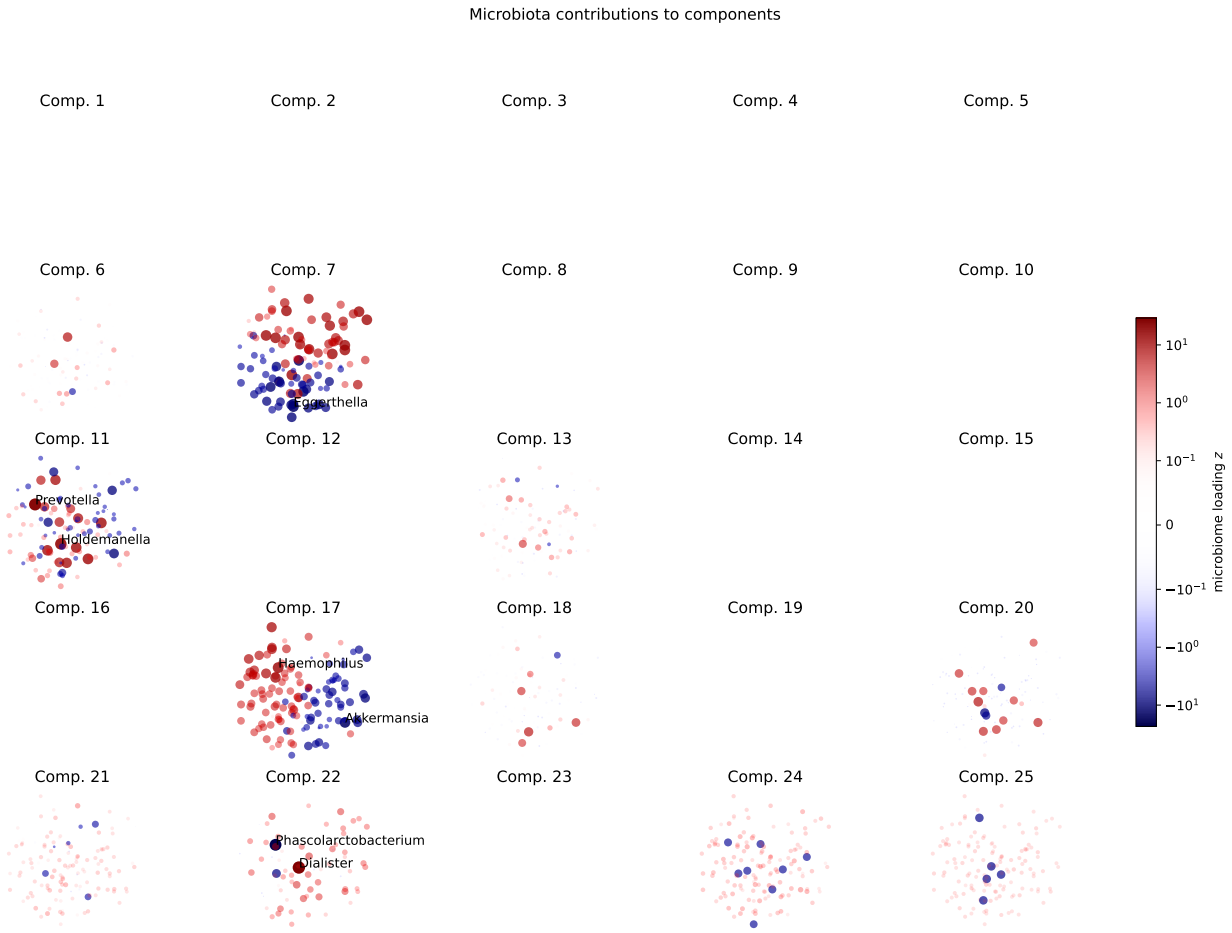


Figure 4: The signed contributions of each microbiota to each of the 25 components. Red indicates positive relations and blue negative. For the most relevant biota the names are also displayed.

2.3 High-Dimensional ALS patient data from the iDPP 2022 Challenge (HIGH-DIM)

This component implements different visualisation techniques in order to efficiently visualise the high dimensional ALS dataset, taken from the Intelligent Disease Progression Prediction (iDPP) challenge.

This component allows the user to select dimensions and visualization types in a straightforward way, supporting mainly the manual data exploration. Because of the simple, but powerful windowing system of the component, the user can be highly creative and flexible in terms of data exploration and visualization. The main focus is not only the simplicity and flexibility, but also the visualization techniques, which are implemented, and which will be implemented in future (see Section 2.3.5). Furthermore, the component could also useful for preparing presentations and summaries of the data.

The challenge description can be found in [Ferro, 2022a]. Different visualisation techniques have been implemented and analyzed, in order to enhance the data exploration experience. These techniques include:

- Scatter Chart (2D)
- Bar Chart
- Scatter Chart Matrix
- Data filtering possibility

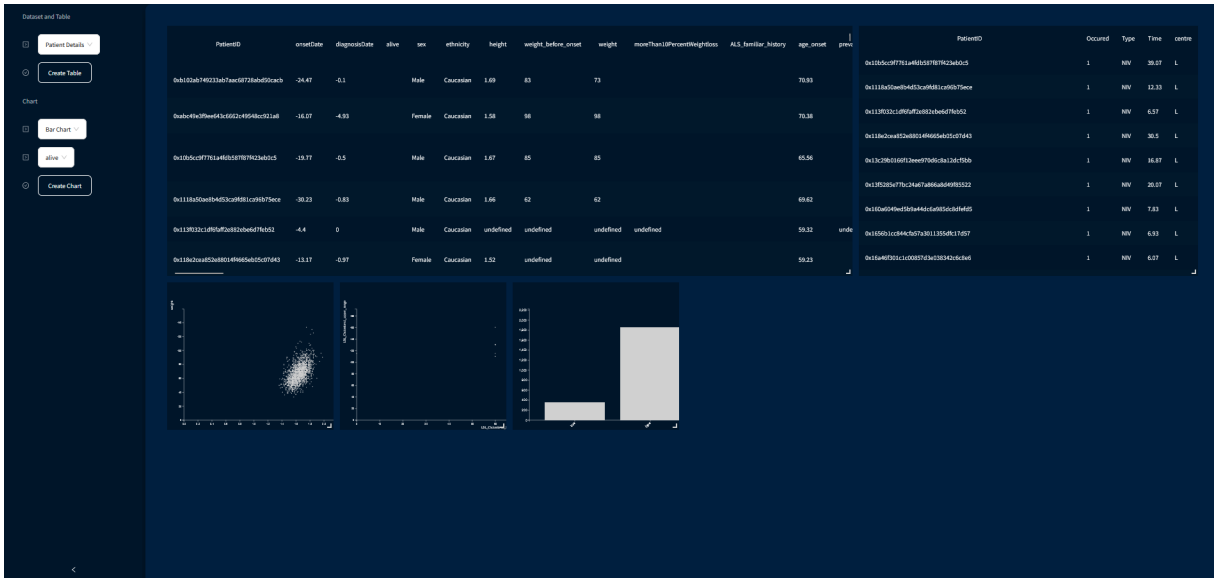


Figure 5: Screenshot of the ALviS system's frontend

2.3.1 Considered Data / Test Dataset / Task Description

The challenge provided three datasets in CSV format:

- Static Variables of Patients: static_vars.csv
- Visit Events of Patients: visits.csv
- Outcome of Patients: outcome.csv

Each dataset includes a column labeled "PatientID," which represents patients using a 16-byte hash. Across the three datasets, there are 2,204 unique patients in total. Some rows in each dataset contain missing values, requiring the development and application of efficient techniques to address these gaps. Detailed descriptions of the datasets are provided in below paragraphs.

Static Variables of Patients This data encompasses all the essential and unchanging details about a patient. This includes for example:

- Diagnosis Date of the disease
- If patient is still alive
- Gender, height, weight
- More detailed parameters: If suffered from stroke, heart disease, etc.

Furthermore, there are in total 98 different columns in this dataset, of which 39 are numerical and 59 are categorical data types.

Visit Events of Patients The patients were visited regularly in the course of their disease and this visits were documented, in order to track disease progression. Some example columns:

- ALSFRS-R questionnaire results
- Information about ongoing therapies
- Date of a therapy start

Explanation of each column can be found in [Ferro, 2022b].

Outcome of Patients This dataset contains the outcomes of the patients. It shows what event occurred to the patient (e.g., needing NIV, PEG, passing away, or no event) and when it happened, measured in months. If no event occurred ("NONE"), it records the time of the last check-up instead. There are only 5 columns in this dataset:

- The PatientID
- If outcome of patient is documented
- What type of event occurred
- The time when the event occurred
- In which research center it happened: Turin or Lisbon

2.3.2 Visual Analytics Approach

The visualisation dashboard can be seen in Figure 5. On the left panel the user has a menu, where they can select different parameters and interact with the system:

- Select one of the 3 datasets
- Create table, using the selected dataset
- Select chart type

- Select columns to render, using the selected chart

For now the user can select between 2 chart types: Bar Chart and Scatter Chart. The visualisation approach is held very general: The user has the possibility for creating multiple chart types and can compare different dataset parameters against each other. This is possible, because the layout is structured as a grid and each visualisation is a new window, which the user can move around and place freely inside the grid layout.

2.3.3 Implementation Description (Backend, Frontend, Development Environment)

This application is structured into a frontend and into a backend. In the following, this is be discussed in more detail.

Frontend The frontend is implemented in TypeScript[3] together with the Vite build system[4]. The frontend framework of choice was React and it was initialized using the template from Vite [5]. In React, the following libraries are additionally used:

- zustand: State management library
- antd: UI components library
- d3: Charting library, to create SVG charts
- react-grid-layout: Offering grid-layout implementation for table and chart visualisation
- react-virtuoso: Virtualized table, to efficiently render large tables

Backend The backend is implemented in the Python programming language. It serves as a REST API endpoint and the server is implemented using FastAPI[6]. The server is processing the dataset requests from the client. So far there are 2 GET requests implemented:

- GET csv/{csv_name}: Returns whole dataset as JSON
- GET csv/{csv_name}/{column}: Returns column in specified dataset as JSON

The following libraries are used in Python:

- FastAPI: Server REST API implementation
- pandas: Process CSV files efficiently

The package manager of choice in the backend is Poetry[7] and is used to track the dependencies for Python.

2.3.4 Demonstration of Preliminary Use Case(s)

Figure 5 shows the current preliminary use case of this system. The user can select between datasets, create multiple tables and charts with different columns and can organize the view according to his needs. Through bar charts, the user can have insight into the general distribution of categorical data. Scatter charts give the user the ability to see correlations between different variables in the datasets. When hovering over the bars or the points, the user can see the exact value.

[3] https://www.typescriptlang.org/
[4] https://vite.dev/
[5] https://vite.dev/guide/#scaffolding-your-first-vite-project
[6] https://fastapi.tiangolo.com/
[7] https://python-poetry.org/

2.3.5 Development Roadmap: What to Develop Next?

In the first place, the next tasks for the system will focus on improving the current functionality. This involves fixing existing bugs and any visual anomalies to provide a stable user experience. Once these issues are resolved, more features will be added. These features include:

- Dimensionality reduction techniques (Principal Component Analysis (PCA), Uniform Manifold Approximation and Projection (UMAP), etc.) to simplify the analysis of high-dimensional datasets.
- More chart types: Heat Map, 3D Scatter Chart, Parallel Coordinates and more.
- Data filtering method, enabling users to filter data based on their specific inputs, hence improving the systems flexibility.
- Extension of data filtering to natural language, to even further alleviate data exploration.
- Implementation of chart interactivity tools: Zooming, Panning, Brushing, Range Selection, etc.
- Different clustering methods, these include: Centroid-based clustering, model-based clustering and density-based clustering.
- Export current workspace as JSON and import again to continue on work. Furthermore, if time constraints allow it, implement session handling and store workspace on server.

2.4 High-Dimensional Patient Data: ALS, PD, MS (HIGH-DIM)

This component focuses on identifying dependencies and groups within the dataset. The dataset includes cognitive, treatment, and demographic data on patients with Parkinson's disease. Understanding how these different attributes influence each other is of great interest, as it can aid in explaining, predicting, and potentially preventing cognitive decline. In this component, we enabled the interactive exploration of patient data, enabling identification of dependencies and patterns across clinical variables using visualizations such as histograms, scatterplots, heatmaps, and PCA biplots. It also facilitates the analysis of whether different test methods potentially introduce data shifts or biases.

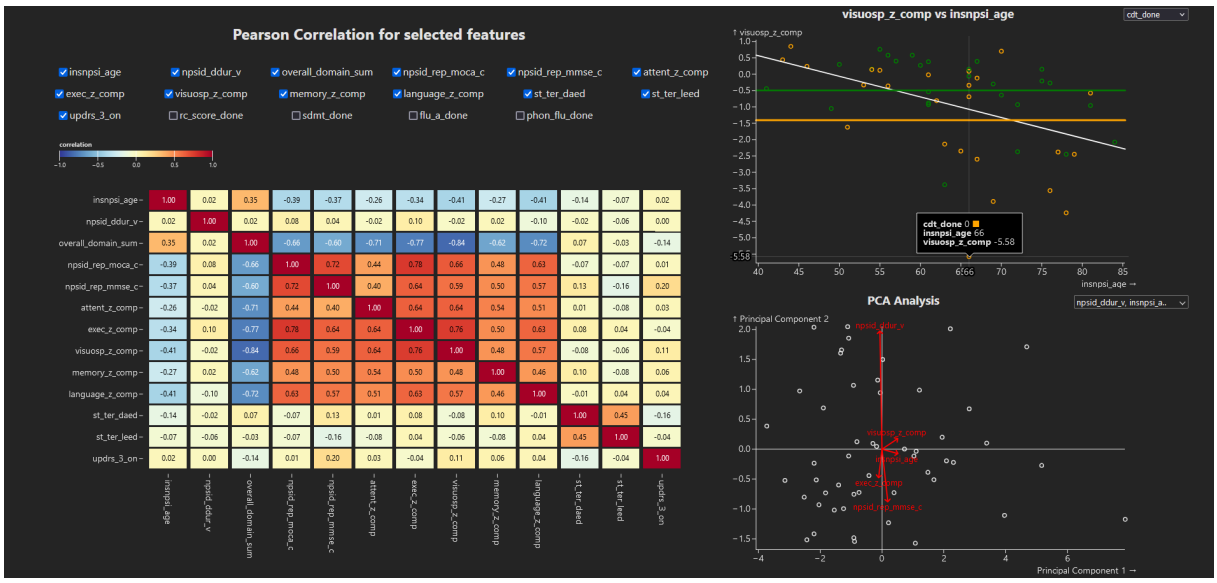


Figure 6: The Dashboard, Correlation Heatmap on the left, Scatterplot and Biplot on the right. In the scatterplot, test methods are color-coded, and the average for each test method is shown.

2.4.1 Considered Data / Test Dataset / Task Description

Initially, three datasets were considered, corresponding to ALS, PD, and MS. However, for this deliverable, only the PD dataset was analyzed.

PD Dataset The dataset comprises approximately 100 features for 50 patients. It includes demographic data such as patient age and duration of PD diagnosis, alongside results from various cognitive tests and treatment details. Several columns contain redundant information, with some measurements represented both numerically and categorically. Additionally, cognitive tests were conducted using multiple methodologies, leading to potential domain shifts. Binary indicators in the dataset denote the use of specific test methods.

After accounting for duplicate information and test method markers, the dataset includes approximately 15 unique measurements.

2.4.2 Visual Analytics Approach

A correlation heatmap, scatterplots, and histograms are used to provide an overview of the data and enable the exploration of variable dependencies. For cognitive z-scores, which were obtained using different test methods, the visualization color-codes these methods to identify potential shifts due to methodological changes. PCA was applied to detect underlying group structures, with results presented as biplots for interpretability.

2.4.3 Implementation Description (Backend, Frontend, Development Environment)

The dashboard was implemented as a *React*[8] application using *TypeScript*. Plots were created with the *observableHQ*[9] and *D3*[10] libraries to enhance interactivity. All calculations were performed in the TypeScript frontend. The dataset's small size (under 100 KB) made this feasible.

2.4.4 Demonstration of Preliminary Use Case(s)

The heatmap and scatterplots enable the identification of linear dependencies, such as the observation that cognitive performance declines with increasing age (*insnpsi_age*), while the duration of PD diagnosis has little effect on test scores. Figure 6 illustrates a scatterplot comparing the *visual z-score* with patient age. It reveals that patients who took the CDT test exhibit different average scores, which could potentially skew the data. This should be accounted for in subsequent analyses. In the PCA biplot, users can select features of interest to visualize their influence on the principal components. Scatterplots and histograms are dynamically generated by clicking on non-diagonal and diagonal cells of the heatmap, respectively, with test methods selectable via a dropdown menu.

2.4.5 Development Roadmap: What to Develop Next?

Future developments will include additional correlation methods, clustering algorithms on PCA data (with cluster visualizations integrated into scatterplots), and incorporating categorical data. The feasibility of an interactive Large Language Model (LLM)-based chatbot for explanation and interpretation will also be explored, along with adapting the dashboard for other datasets. This could assist analysts unfamiliar with certain dataset variables.

[8] https://react.dev/
[9] https://observablehq.com/documentation/cells/observable-javascript
[10] https://d3js.org/

2.5 Clusters in Focus: Detail-On-Demand Analysis Dashboard (HIGH-DIM)

This component provides a structured approach for analyzing high-dimensional biomarker data through interactive exploration of feature pairs, clustering results, and similarity metrics, enabling the identification of indicative biomarker patterns.

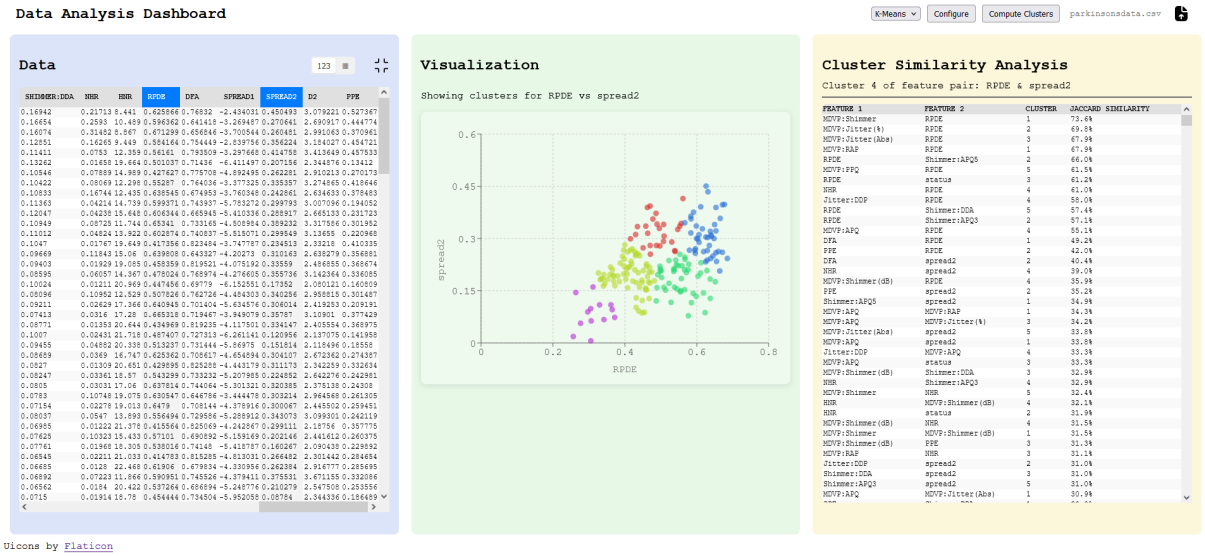


Figure 7: Overview of the component, integrating feature-level exploration, clustering, and similarity analysis.

2.5.1 Considered Data / Test Dataset / Task Description

The dataset used during development consists of biomedical voice measurements collected from 31 individuals, 23 of whom have Parkinson's disease. This dataset was sourced from [Little et al., 2009]. It includes 24 features and one identifier column, with a total of 195 data points. The features exhibit varying degrees of correlation, which is beneficial as it highlights the utility of different views within the system. For example, the heatmap [9] view helps users visually identify likely correlations between features without requiring detailed statistical analysis.

The primary goal of the system is to enable users to explore high-dimensional biomedical data and identify potential relationships between biomarkers that may have diagnostic significance. The workflow includes feature exploration, clustering visualization, and similarity analysis to reveal consistent patterns and correlations.

2.5.2 Visual Analytics Approach

The component consists mainly of three different panels which dynamically expand and shrink based on the current stage in the workflow:

- Panel 1: Feature Exploration (8a): Users can explore the dataset on a granular level, with options to view raw numerical data (8a), heatmaps of feature correlations (9), and histograms of individual feature

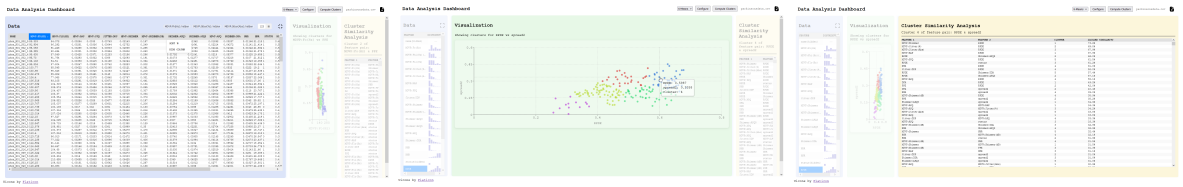
distributions (10). The heatmap view is particularly effective in identifying potential feature correlations visually, while the histogram view offers interactive customization for deeper analysis. Feature pairs for further clustering can be selected directly from this panel.

- Panel 2: Clustering Visualization (8b): Displays clustering results for the selected feature pair as an interactive plot. Users can click on clusters to examine their composition and proceed to similarity analysis in the next panel.

- Panel 3: Cluster Similarity Analysis (8c): Shows clusters from other feature pairs ranked by Jaccard Index Percentage:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

where A and B are the respective clusters being compared. This analysis helps identify different feature pairs that lead to similar clustering, providing potential insights into biomarker relationships.



(a) Tabular / numerical view of selectable dataset features.

(b) Cluster visualization for a feature pair selected previously.

(c) Similarity-ranked clusters across feature pairs.

Figure 8: The three-layered view architecture of Clusters in Focus: Starting from the first view, one can navigate to the second and third based on selections made in the process.

Figure 9: Heatmap view showing feature correlations.

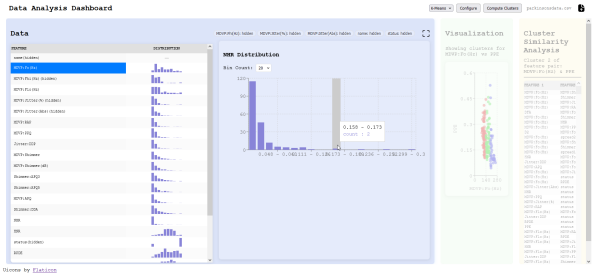


Figure 10: Histogram view of feature distributions.

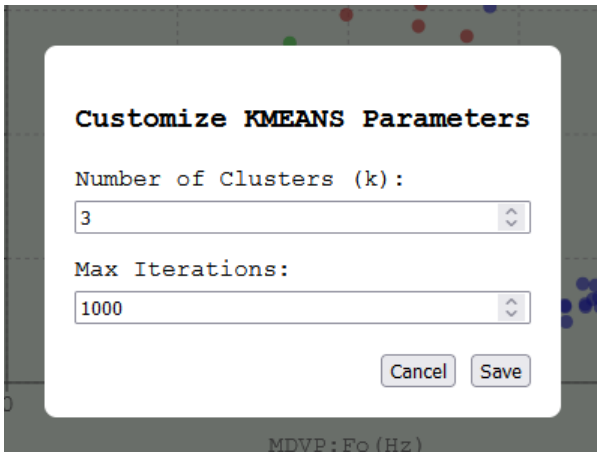


Figure 11: Clustering parameters customization interface.

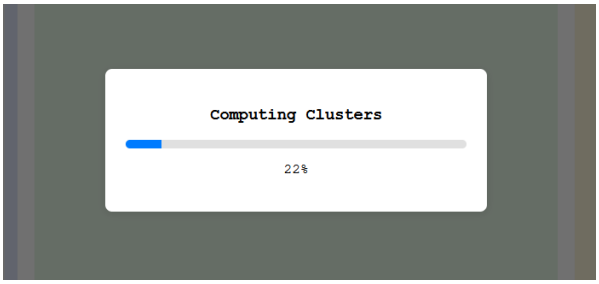


Figure 12: Clustering computation progress visualization.

2.5.3 Implementation Description (Backend, Frontend, Development Environment)

Frontend: The entire system, including computation and visualization, is implemented in the frontend using Vite and React for high-performance, interactive interfaces.

Cluster Computation: Clustering is performed directly in the browser as a blocking operation upon user request (12) using a pure TypeScript implementation [11]. Results are stored in the browser's local storage, avoiding recomputation unless parameters change.

Clustering Algorithms: Currently supports KMeans (11) with customizable parameters. Computations are optimized for small to moderate dataset sizes, with plans to improve scalability for larger datasets.

2.5.4 Demonstration of Preliminary Use Case(s)

A typical use case involves:

Step 1: Dataset Upload and Exploration. The user uploads a dataset of voice measurements, including features like *PPE* (Pitch Period Entropy) and *MDVP:Fo(Hz)* (Fundamental Frequency). These features are potential biomarkers for Parkinson's disease, with *PPE* reflecting pitch irregularity and *MDVP:Fo(Hz)* capturing vocal frequency. In Panel 1, the user explores feature relationships visually (e.g., heatmaps, histograms) and selects feature pairs for clustering.

Step 2: Clustering Analysis. The user selects the pair *MDVP:Fo(Hz)* and *MDVP:Shimmer(dB)*, clustering it using KMeans (4 clusters, max 1000 iterations). Panel 2 visualizes the clustering, showing groups of datapoints with similar feature values, potentially reflecting disease progression.

[11] https://www.npmjs.com/package/ml-kmeans

Step 3: Similarity Analysis Across Feature Pairs. In Panel 3, Cluster 1 from *MDVP:Fo(Hz)* and *MDVP:Shimmer(dB)* is analyzed for similar clusters across other feature pairs, ranked by Jaccard similarity. Examples include: - ***MDVP:Fo(Hz)* and *MDVP:RAP*:** Cluster 3 (100.0% similarity), showing alignment between vocal frequency and jitter-related features. - ***MDVP:Fhi(Hz)* and *Shimmer:APQ5*:** Cluster 4 (87.3%), highlighting connections between high vocal frequency and amplitude variation. - ***MDVP:Flo(Hz)* and *MDVP:Fo(Hz)*:** Cluster 1 (81.3%), aligning low and average vocal frequencies.

Interpretation of Results. These clusters suggest consistent relationships between vocal frequency, amplitude variation, and clinical status. For example, *Shimmer:DDA* and *MDVP:RAP* align closely with frequency-based clusters, reinforcing their value as biomarkers. This workflow enables the discovery of alternative biomarker combinations, such as using *MDVP:Fhi(Hz)* and *Shimmer:APQ5*, for diagnosis or monitoring.

2.5.5 Development Roadmap: What to Develop Next?

Potential future work includes:

- **Scalability Enhancements:** For larger datasets, enable dynamic data loading and incorporate heuristics to skip irrelevant feature pair clusterings.
- **Feature Set Customization:** Allow ad hoc filtering or exclusion of features from clustering.
- **Ephemeral PCA Columns:** Add the ability to create and include PCA-transformed features in the analysis.
- **Additional Clustering Algorithms:** Integrate clustering methods such as Gaussian Mixture Models (GMM), DBSCAN, and Spectral Clustering to support diverse data structures and clustering needs.
- **n-Tuple Feature Clustering:** Extend clustering analysis to n-dimensional feature combinations with embedding algorithms like Spectral Clustering.
- **Statistical Correlation Analysis:** Integrate automatic statistical measures to identify significant feature relationships.
- **Preprocessing Options:** Enable on-the-fly data normalization, outlier handling, and other preprocessing steps.

2.6 Multi-Component State Model Visualisation (HIGH-DIM, TIME)

This component is concerned with the visualisation of a multi-component state model coupled (biological) system. Each component can be modelled through its own individual state model, including dependencies on the environment such as ion concentrations, voltages or other variables.

This component builds on and extends open code which was initially developed as part of a stipend by Peter Waldert with the BioTechMed Graz research center[12] within its Lab Rotation program [13]. The work in HEREDITARY extends the existing component by time series and marker designs. The integrated component can be used in a versatile way to visualize large volumes of multivariate time series data.

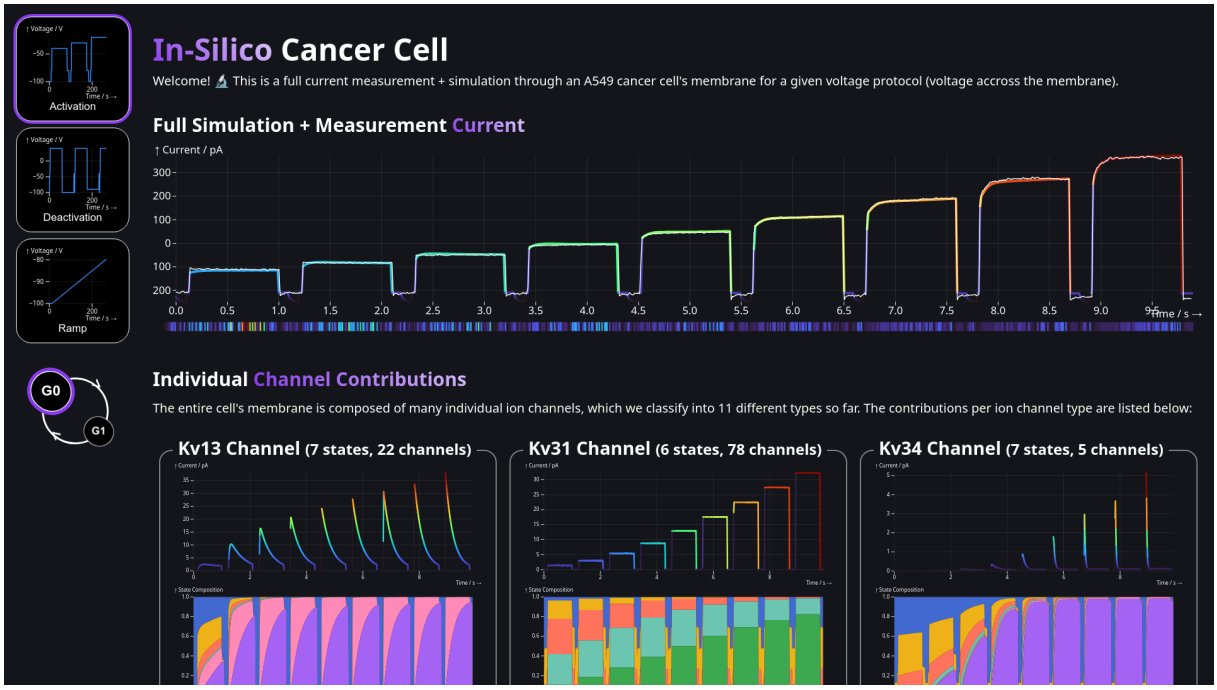


Figure 13: Screenshot of the live simulation dashboard, available here: https://in-silico.hce.tugraz.at/. This is the graphical user interface to the simulation written in Rust. The full simulation, completing within around 500ms, runs live in the browser using Rust's bindings to WebAssembly. One can choose the voltage protocol and cell cycle phase (G0/G1) in the menu on the left, while the full current and individual channel currents and internal states can be seen on the right. The remaining ion channel types have been cut off on this screenshot.

2.6.1 Considered Data / Test Dataset / Task Description

The present visualisation (Figure 13) takes timeseries data of the environmental factors and the state model itself as input. With the current implementation, the specification of a simple transition matrix (function of the environment) is enough to couple and simulate the system. The visualisation then shows individual component contributions to the whole, over time, with a clear relationship to the component's state. In this

[12] https://biotechmedgraz.at/de/
[13] https://biotechmedgraz.at/en/biotechmed-graz/programs/lab-rotation-program/

exmaple, we work with the A549 model introduced in [Langthaler et al., 2021, 2024]. Physical measurements are obtained using a *Patch-Clamp System*, where one records the current through the membrane given a voltage protocol.

2.6.2 Visual Analytics Approach

We introduce a visualisation approach of the entire multi-component model: a live simulation dashboard available online, running directly in the browser. The goal behind this in-browser environment is to make the topic as accessible as possible, enabling further steps in the direction of science communication and dissemination, aligned with the HEREDITARY goals.

For our given example of the A549 cell model, the system input is a voltage protocol $V(t)$ and the observable quantity is a current $I(t)$ (cf. Section 2.6.4 below). In that instance, the whole cell current $I : T \to \mathbb{R}$ over time $t \in T \subset \mathbb{R}^+$ is the sum of all individual channel contributions $I_k, k \in \{1, ..., M\}$ over $M \in \mathbb{N}$ channel types

$$I(t) := \sum_{k=1}^{M} N_k I_k(t) = \sum_{k=1}^{M} N_k g_k p_{o,k} \left(V(t) - E_k\right) .$$

At each time step, the next state $\boldsymbol{s}_{k,n+1} \in [0,1]^{N_{s,k}}$ of the k-th channel type is obtained by

$$\boldsymbol{s}_{k,n+1} = H_k \left(V(t_n), \boldsymbol{C}(t_n), t_n\right) \boldsymbol{s}_{k,n} , \quad \text{with} \quad t_n := \sum_{i=0}^{n} (\Delta t)_i .$$

In order to accelerate the simulation in areas where there is little change to the dynamics, we choose an adaptive step size $(\Delta t)_n$ based on

$$(\Delta t)_{n+1} = (\Delta t)_n \left(\frac{\Delta^{\text{tol}}}{\sum_{k=1}^{M} N_k \| \boldsymbol{s}_{k,n+1} - \boldsymbol{s}_{k,n} \|} \right)^{1/2} .$$

with an external parameter $\Delta^{\text{tol}} = 10^{-4}$ that governs the state change tolerance in between simulation steps. It is chosen using a systematic parameter sweep.

2.6.3 Implementation Description (Backend, Frontend, Development Environment)

Due to the computational load of the simulation at hand, the forward iteration itself is implemented in Rust, a low-level programming language which supports compilation to WebAssembly (WASM). The system is therefore compatible with both, standard, platform-dependent binary execution and a browser environment. The iterative simulation runs with adaptive timestepping and a highly efficient implementation in the Rust programming language, which supports compilation to these multiple targets.

A key ingredient to the performance of the simulation is to statically encode the state vector sizes, making them fully known at compile-time. This allows the compiler to construct a highly efficient version of the binary ahead of time, providing the user with a fluent usage experience. This process may be further enhanced by utilising Performance-guided optimisation (PGO), which analyses frequently called subsections of the code from preliminary tests and optimises the call order and code layout in memory.

The corresponding inverse problem to find parameters to the model is expressed as a quadratic program, and can thereby be solved within milliseconds.

2.6.4 Demonstration of Preliminary Use Case(s)

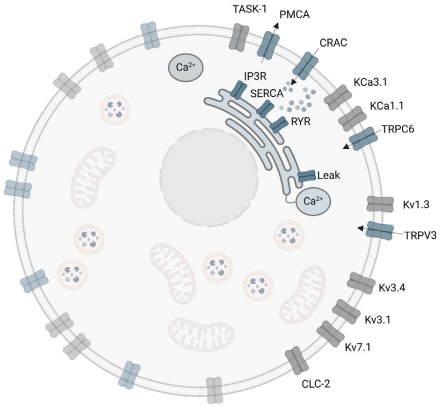


Figure 14: The extended A549 cell model [Langthaler et al., 2024].

In order to model the electrophysiological behaviour of a cell, and therefore its response to changes in its environment, it is crucial to develop an understanding of its membrane. A cell's membrane consists of multiple individual ion channels, which allow for the flow of charges, such as Kalium, Calcium or Chlorine, through the membrane. Physiologically, channels are either open or closed, usually based on the relative positioning of protein boundaries within the channel. In the A549 model, this positioning is represented through a state model, where each individual state represents the current position of the protein within the channel. However, we can only observe whether a channel is open or closed at any given time, therefore making this a Hidden Markov Model (HMM).

2.6.5 Development Roadmap: What to Develop Next?

The coupled, multi-component state model visualisation approach leads to a novel basis for the visualisation of such complex, computationally expensive systems. Efforts to further enhance and complete the first electrophysiological cancer cell model simulation were fruitful, resulting in a new Python library implementation, an improved inverse problem solution technique and a live in-browser simulation dashboard. Due to the computational nature of the problem, the adaptation of the visualisation to further systems is non-trivial and requires some knowledge of the Rust programming language. This will also be subject to further efforts to make the tool more accessible, on top of the existing Python library and in-browser environment. There are many ideas for further customizations that would improve the live simulation dashboard, opening up future usage perspectives.

2.7 Ontology and Sparse data Exploration Tool (OnSET) (NETWORK, HIGH-DIM, TEXT)

The OnSET (Figure 15) enables a two-step exploration of arbitrary ontologies in combination with related sparse data used to either inform the ontologies or gathered separately and being applied to the ontology analysed.

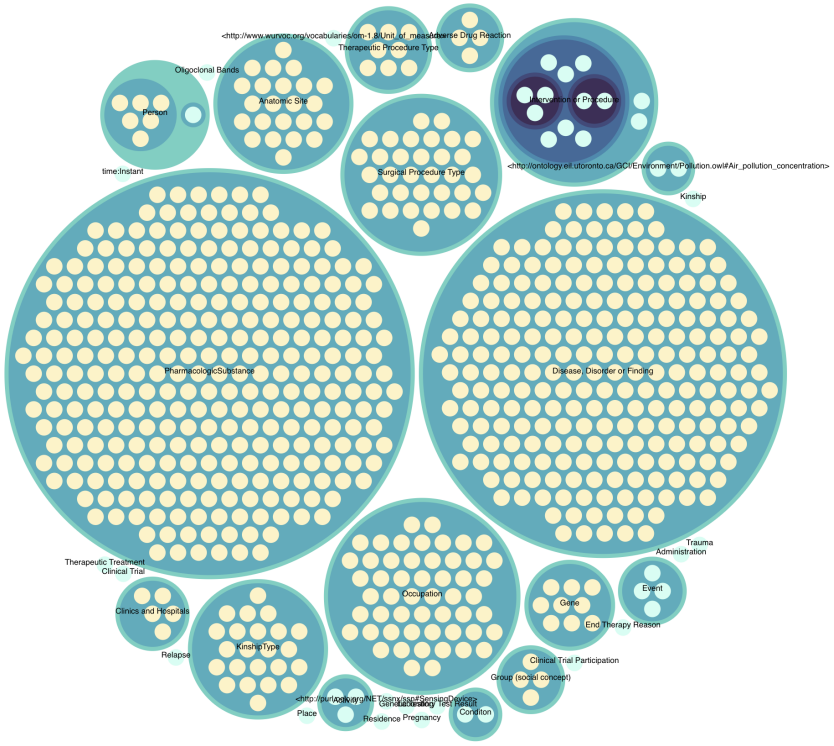


Figure 15: Overview of the HEREDITARY phenoclinical ontology (HERO) within OnSET

2.7.1 Considered Data / Test Dataset / Task Description

While OnSET is capable of supporting any ontology and sparse dataset combination, we use the HERO for testing and evaluating the tool. The HERO builds upon the Bringing Artificial Intelligence home for a better care of amyotrophic lateral sclerosis and multiple sclerosis (BRAINTEASER) Ontology [Faggioli et al., 2024b] and extends it towards concepts and instances present in the HEREDITARY project. The ontology relates attributes of patients with different diseases and diagnoses, and clinical procedures.

The sparse dataset as the basis for the linked visualization is the related BRAINTEASER ALS and MS dataset [Faggioli et al., 2024a], which could also be exchanged for another sparse dataset relating to an arbitrary loaded ontology. This relation of the data is achieved using text-based matching of concepts presents in either dataset.

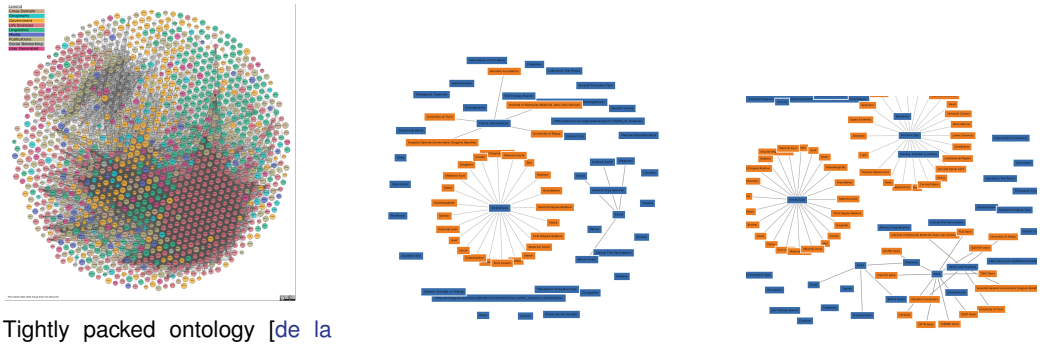
These two data sources are then used to build a visualization tool to aid users with little to no experience in ontologies in

1. the explorative overview of arbitrary ontologies for users not familiar with the usual notation of ontologies, and
2. the relation (co-occurrence) of specific instances, their properties and classes given a sparse dataset.

While this tool's target audience is mainly non-experts, we also intend the tool to be used as an initial overview of ontologies and as a point of entry for the different aspects and ideas within HEREDITARY.

2.7.2 Visual Analytics Approach

Principal visualization and interaction techniques Ontologies are usually visualized using different graph placement strategies like manual placement, tight packing or predetermined shapes like circles, as shown in Figure 16.



(a) Tightly packed ontology [de la Cueva and Méndez, 2022]

(b) Force Layout Graph

(c) Overflowing Graph

Figure 16: Different Ontology Visualizations

These are the reasons why we choose the circle packing approach as the core principle for OnSET. Circle packing enables an explorative and hierarchical view of the ontologies [Bostock et al., 2011; Wang et al., 2006]. This approach can, compared to traditional network visualizations operate intuitively on multiple scales and provide the users with a complete overview of the data while providing details on demand while still maintaining an overview of the grander structure.

The circle packing, while utilizing the space efficiently, still allows for additional relational visualization on top as the circle can be seen as the backdrop for the links among them.

Preliminary requirement / task description The OnSET overview should enable a two-step learning process for users with little to no experience in the ontologies they are exploring by first enabling a rough overview of the different hierarchies present in the dataset, and in the next step relating these hierarchies between each other based on the presence in sparse datasets.

Visual analytics design This visualization uses arbitrary ontologies and maps the classes, subclasses and named instances into the overview using a hierarchical representation of the ontology. This mapping enables the overview of Figure 15. The user can, based on interest in specific topics, click on any level within the circle packing to zoom and pan to reveal more details about the individual subclasses and instances of a class as shown in Figure 17.

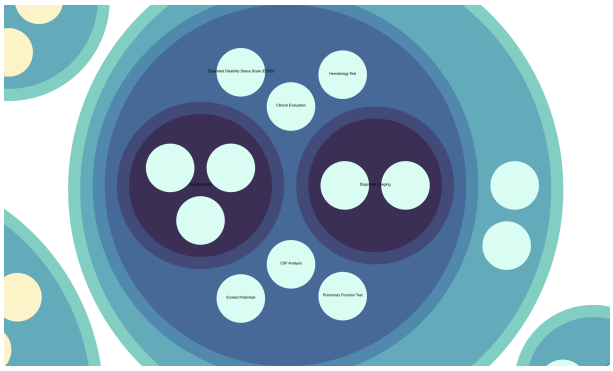
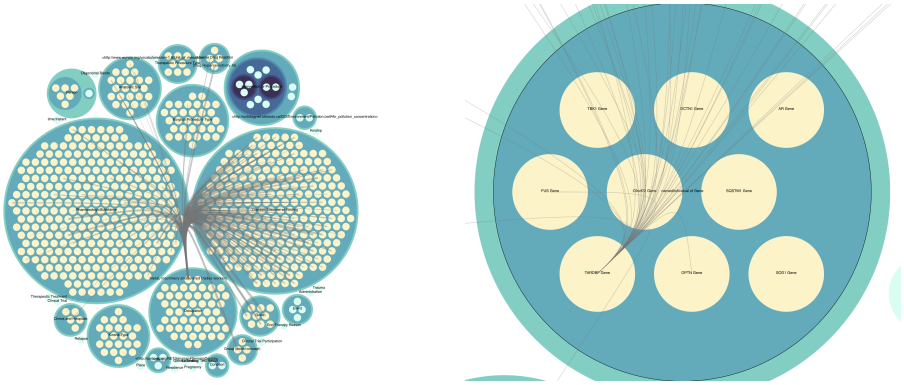


Figure 17: Detailed class view of HERO within OnSET

The visualization furthermore uses related sparse data sources and maps the data to the ontologies, providing co-occurrences and relations across the ontology graph. These co-occurrences are then shown on-demand by clicking on a specific node. These links inform the user both in the overview level and the detail views, both shown in Figure 18.



(a) Links spanning the whole visualization **(b)** Links within a small detail view

Figure 18: Co-occurrences displayed as links in OnSET

The overview level of Figure 18a enables a top-level estimation of the co-occurrences across the whole ontology. This enables the user to inform themselves about related concepts within the data. The detailed link view shown in Figure 18b fosters this learning within the selected levels. This is helpful if the relations occur locally and might be missed at the global scale.

2.7.3 Implementation Description (Backend, Frontend, Development Environment)

The described visual analytics features are achieved using a three-step system architecture illustrated in Figure 19. This system is based upon an Extraction, Transform and Load (ETL) pipeline, where we first extract the datasets and ontologies. The extraction step aims to consolidate similar datasets into one table for the tabular data, and extract the major relationships from the ontologies. Both the table schema and ontology information are then embedded using the `sentence-transformer` package [Reimers and Gurevych, 2020].

This textual translation approach enables us to merge two unrelated schemas with a good accuracy and provides intuitive matches. We refine these matches by ranking them and assigning them using a greedy heuristic. These reduced matches are then loaded into a SQLite[14] database and served, along the ontology itself, on our FastAPI[15] backend and visualized using our Vue[16] frontend with the help of *D3*[17].

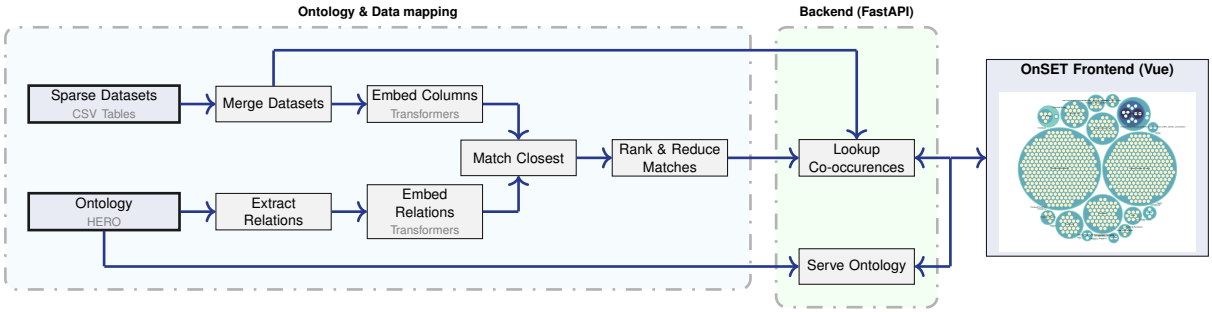


Figure 19: OnSET architecture

2.7.4 Demonstration of Preliminary Use Case(s)

The deployed architecture enables the explorative tasks already outline in Section 2.7.2 to foster understanding of concepts in sparse datasets using an informed ontology as the reference. While we demonstrate the capabilities of the OnSET visualization using the HERO, the visualization is not limited to this ontology and can load arbitrary ontologies and sparse datasets to support other fields of interest.

The tool could support a workflow, where a researcher has acquired an ontology for a new field of interest and would like to know more about the hierarchical layout of the ontology. The ontology can then be loaded into OnSET and explored, i.e. look for concepts of interest and check whether it even matches their interest. If the researcher additionally gathers related tabular data, the tool can load it and try to match it to the ontology. This connection of schematic data and tabular data allows the researcher to look for relations within their data using the existing ontologies.

2.7.5 Development Roadmap: What to Develop Next?

We will explore multiple future avenues within the OnSET system for the next steps in our project, like further improvements to the layouting of the components, considerations of stakeholder feedback and explorations on the applicability of other datasets.

First, the OnSET system will be discussed with domain stakeholders within HEREDITARY in the next steps to implement more of their specific requirements. The goal of these talks will be the incorporation of feedback for this initial exploration and collection of suggestions and directions within the project. These directions should influence our decisions for further improvements to the layouting approach.

We will furthermore explore different datasets from different domains, like the EuroVoc [European Union, 2024] ontology and connect these with related data, like the European Parliament plenary sittings to explore the capabilities of the tool in text-heavy settings. Another direction for OnSET is the possibility to provide a

[14] https://www.sqlite.org/
[15] https://fastapi.tiangolo.com/
[16] https://vuejs.org/
[17] https://d3js.org/

point of entry towards HEREDITARY, using the ontology, to link to other systems and visualization tools based on the domain of interest of the end users, with the possibility to suggest linked topics and visualizations exploring these links.

3 Conclusion and Next Steps

Based on initial requirement analysis and example data from HEREDITARY partners and use cases, we have developed a suite of visualization components. The components support key data types of interest in HEREDITARY (Network, Time-Dependent, High-Dimensional, and Text-based Data). Complementary components (see Deliverable 5.3) support spatial, image data and simulation data.

As next important steps, we will build dedicated visual analytics applications to support research problems and use cases of the HEREDITARY research partners. To this end, dedicated requirement analysis will inform the application development and evaluation (Task 5.5: Requirements analysis, evaluation, and user studies). The applications will support detailed user interaction, specifically, visual querying and analytical interaction (Task 5.3: Visual querying and analytical interaction). Also, system support for guiding users through complex data is a research priority (Task 5.4: User guidance approaches).

The obtained results will also be presented to the biomedical visualization research community. A first result on the DROPLETS component (see deliverable 5.3) was presented as a technical report at the BioMedvis Challenge Workshop at IEEE VIS 2024[18] The work received much attention and a challenge award for novelty. Work is ongoing to develop the visualization into a fully interactive system, and apply it to further spatial data types. As part of our research, we will contact researchers in the area, and discuss our components and further development possibilities. And mainly, we will develop visual analytics applications for HEREDITARY use cases.

[18] Stefan Lengauer, Peter Waldert, Tobias Schreck: Droplets: A Marker Design for visually enhancing Local Cluster Association. Report at the Bio+MedVis Challenge, co-located with IEEE VIS 2024.

References

Aigner, W., Miksch, S., Schumann, H., and Tominski, C. (2023). *Visualization of Time-Oriented Data, Second Edition*. Human-Computer Interaction Series. Springer.

Beck, F., Burch, M., Diehl, S., and Weiskopf, D. (2017). A taxonomy and survey of dynamic graph visualization. *Comput. Graph. Forum*, 36(1):133–159.

Bostock, M., Ogievetsky, V., and Heer, J. (2011). D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309.

Cakmak, E., Schlegel, U., Jäckle, D., Keim, D. A., and Schreck, T. (2021). Multiscale snapshots: Visual analysis of temporal summaries in dynamic graphs. *IEEE Trans. Vis. Comput. Graph.*, 27(2):517–527.

de la Cueva, J. and Méndez, E. (2022). *Open Science and Intellectual Property Rights. How Can They Better Interact? State of the Art and Reflections. Report of Study. European Commission.* European Commission, Brussels.

European Union (2024). Eurovoc - eur-lex. https://eur-lex.europa.eu/browse/eurovoc.html. (Accessed on 11/14/2024).

Faggioli, G., Marchesin, S., Menotti, L., Aidos, H., Bergamaschi, R., Birolo, G., Bosoni, P., Cavalla, P., Chiò, A., Dagliati, A., de Carvalho, M., Di Nunzio, G. M., Fariselli, P., García Dominguez, J. M., Gromicho, M., Guazzo, A., Longato, E., Madeira, S. C., Manera, U., Silvello, G., Tavazzi, E., Tavazzi, E., Trescato, I., Vettoretti, M., Di Camillo, B., and Ferro, N. (2024a). Brainteaser als and ms datasets.

Faggioli, G., Marchesin, S., Menotti, L., Nunzio, G. M. D., Silvello, G., and Ferro, N. (2024b). The brainteaser ontology for als and ms clinical data.

Fekete, J., van Wijk, J. J., Stasko, J. T., and North, C. (2008). The value of information visualization. In Kerren, A., Stasko, J. T., Fekete, J., and North, C., editors, *Information Visualization - Human-Centered Issues and Perspectives*, volume 4950 of *Lecture Notes in Computer Science*, pages 1–18. Springer.

Ferro, N. (2022a). iDPP @ CLEF 2022 Guidelines - Intelligent Disease Progression Prediction. https://brainteaser.dei.unipd.it/challenges/idpp2022/. (Accessed on 11/15/2024).

Ferro, N. (2022b). iDPP @ CLEF 2022 Guidelines - Visits Explanation. https://brainteaser.dei.unipd.it/challenges/idpp2022/assets/other/visits.txt. (Accessed on 11/15/2024).

Gerrits, T., Czappa, F., Banesh, D., and Wolf, F. (2024). IEEE SciVis Contest 2023 - Dataset of Neuronal Network Simulations of the Human Brain.

Gillmann, C., Saur, D., Wischgoll, T., and Scheuermann, G. (2021). Uncertainty-aware Visualization in Medical Imaging - A Survey . *Computer Graphics Forum*.

Keim, D., Kohlhammer, J., Ellis, G., and Mansmann, F., editors (2010). *Mastering the Information Age: Solving Problems with Visual Analytics*. VisMaster, http://www.vismaster.eu/book/.

Kohn, N., Szopinska-Tokov, J., Llera Arenas, A., Beckmann, C., Arias-Vasquez, A., and Aarts, E. (2021). Multivariate associative patterns between the gut microbiota and large-scale brain network connectivity. *Gut Microbes*, 13(1):2006586.

Kozlíková, B., Krone, M., Falk, M., Lindow, N., Baaden, M., Baum, D., Viola, I., Parulek, J., and Hege, H. (2017). Visualization of biomolecular structures: State of the art revisited. *Comput. Graph. Forum*, 36(8):178–204.

Kreiser, J., Meuschke, M., Mistelbauer, G., Preim, B., and Ropinski, T. (2018). A survey of flattening-based medical visualization techniques. *Comput. Graph. Forum*, 37(3):597–624.

Kucher, K. and Kerren, A. (2019). Text visualization revisited: The state of the field in 2019. In Pereira, J. M. and Raidou, R. G., editors, *21st Eurographics Conference on Visualization, EuroVis 2019 - Posters, Porto, Portugal, June 3-7, 2019*, pages 29–31. Eurographics Association.

Langthaler, S., Rienmüller, T., Scheruebel, S., Pelzmann, B., Shrestha, N., Zorn-Pauly, K., Schreibmayer, W., Koff, A., and Baumgartner, C. (2021). A549 in-silico 1.0: A first computational model to simulate cell cycle dependent ion current modulation in the human lung adenocarcinoma. *PLoS Comput. Biol.*, 17(6):e1009091.

Langthaler, S., Zumpf, C., Rienmüller, T., Shrestha, N., Fuchs, J., Zhou, R., Pelzmann, B., Zorn-Pauly, K., Fröhlich, E., Weinberg, S. H., and Baumgartner, C. (2024). The bioelectric mechanisms of local calcium dynamics in cancer cell proliferation: an extension of the A549 in silico cell model. *Front. Mol. Biosci.*, 11:1394398.

Lawonn, K., Smit, N. N., Bühler, K., and Preim, B. (2018). A survey on multimodal medical data visualization. *Comput. Graph. Forum*, 37(1):413–438.

Little, M. A., McSharry, P. E., Hunter, E. J., Spielman, J., and Ramig, L. O. (2009). Suitability of dysphonia measurements for telemonitoring of parkinson's disease. *IEEE Transactions on Biomedical Engineering*, 56(4):1015–1022.

Miksch, S. and Aigner, W. (2014). A matter of time: Applying a data-users-tasks design triangle to visual analytics of time-oriented data. *Comput. Graph.*, 38:286–290.

Munzner, T. (2014). *Visualization Analysis and Design*. A.K. Peters visualization series. A K Peters.

Oeltze-Jafra, S., Meuschke, M., Neugebauer, M., Saalfeld, S., Lawonn, K., Janiga, G., Hege, H., Zachow, S., and Preim, B. (2019). Generation and visual exploration of medical flow data: Survey, research trends and future challenges. *Comput. Graph. Forum*, 38(1):87–125.

Pretorius, A. J., Khan, I. A., and Errington, R. J. (2017). A survey of visualization for live cell imaging. *Comput. Graph. Forum*, 36(1):46–63.

Reimers, N. and Gurevych, I. (2020). Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Rinke, S., Butz-Ostendorf, M., Hermanns, M.-A., Naveau, M., and Wolf, F. (2018). A scalable algorithm for simulating the structural plasticity of the brain. *J. Parallel Distrib. Comput.*, 120:251–266.

Sacha, D., Stoffel, A., Stoffel, F., Kwon, B. C., Ellis, G. P., and Keim, D. A. (2014). Knowledge generation model for visual analytics. *IEEE Trans. Vis. Comput. Graph.*, 20(12):1604–1613.

Sedlmair, M., Meyer, M. D., and Munzner, T. (2012). Design study methodology: Reflections from the trenches and the stacks. *IEEE Trans. Vis. Comput. Graph.*, 18(12):2431–2440.

Thomas, J. J. and Cook, K. A. (2005). *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Ctr.

Tominski, C. and Schumann, H. (2020). *Interactive Visual Data Analysis*. AK Peters Visualization Series. CRC Press.

van Wijk, J. J. (2005). The value of visualization. In *16th IEEE Visualization Conference, IEEE Vis 2005, Minneapolis, MN, USA, October 23-28, 2005, Proceedings*, pages 79–86. IEEE Computer Society.

von Landesberger, T., Kuijper, A., Schreck, T., Kohlhammer, J., van Wijk, J. J., Fekete, J., and Fellner, D. W. (2010). Visual analysis of large graphs. In Hauser, H. and Reinhard, E., editors, *31st Annual Conference of the European Association for Computer Graphics, Eurographics 2010 - State of the Art Reports, Norrköping, Sweden, May 3-7, 2010*, pages 37–60. Eurographics Association.

Wang, W., Wang, H., Dai, G., and Wang, H. (2006). Visualization of large hierarchical data by circle packing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, page 517–520, New York, NY, USA. Association for Computing Machinery.

Ward, M. O., Grinstein, G. G., and Keim, D. A. (2010). *Interactive Data Visualization - Foundations, Techniques, and Applications*. A K Peters.