Bioinformatics, 34(8), 2018, 1433–1435
doi: 10.1093/bioinformatics/btx760
Advance Access Publication Date: 24 November 2017
Applications Note

Databases and ontologies

FUn: a framework for interactive visualizations of large, high-dimensional datasets on the web

Daniel Probst* and Jean-Louis Reymond*

Department of Chemistry and Biochemistry, National Center for Competence in Research NCCR TransCure, University of Berne, 3012 Berne, Switzerland

*To whom correspondence should be addressed.

Associate Editor: Jonathan Wren

Received on June 12, 2017; revised on November 10, 2017; editorial decision on November 19, 2017; accepted on November 23, 2017

Abstract

Motivation: During the past decade, big data have become a major tool in scientific endeavors. Although statistical methods and algorithms are well-suited for analyzing and summarizing enormous amounts of data, the results do not allow for a visual inspection of the entire data. Current scientific software, including R packages and Python libraries such as ggplot2, matplotlib and plot.ly, do not support interactive visualizations of datasets exceeding 100 000 data points on the web. Other solutions enable the web-based visualization of big data only through data reduction or statistical representations. However, recent hardware developments, especially advancements in graphical processing units, allow for the rendering of millions of data points on a wide range of consumer hardware such as laptops, tablets and mobile phones. Similar to the challenges and opportunities brought to virtually every scientific field by big data, both the visualization of and interaction with copious amounts of data are both demanding and hold great promise.
Results: Here we present FUn, a framework consisting of a client (Faerun) and server (Underdark) module, facilitating the creation of web-based, interactive 3D visualizations of large datasets, enabling record level visual inspection. We also introduce a reference implementation providing access to SureChEMBL, a database containing patent information on more than 17 million chemical compounds.
Availability and implementation: The source code and the most recent builds of Faerun and Underdark, Lore.js and the data preprocessing toolchain used in the reference implementation, are available on the project website (http://doc.gdb.tools/fun/).
Contact: daniel.probst@dcb.unibe.ch or jean-louis.reymond@dcb.unibe.ch

1 Introduction

Here we present FUn, a framework facilitating the development of web applications enabling real-time, interactive rendering of millions of data points on a wide range of devices. As an application example we introduce a publicly accessible reference implementation in form of a 2D/3D visualization of the chemical space spanned by the 17 million molecules from the patent literature as collected in the SureChEMBL database (Papadatos *et al.*, 2016). This implementation enables us to scale up and generalize the approach of our recently reported 3D-chemical space visualization tools WebDrugCS and WebMolCS, which were limited to fewer than 10 000 molecules, to millions of molecules (Awale and Reymond, 2016; Awale *et al.*, 2017; Ruddigkeit *et al.*, 2013). The evaluation of other available solutions for web-based big data visualization has shown that these solutions were either limited in the total amount of data points being rendered or unable to depict point clouds with more than two spatial dimensions. Indeed, available solutions do not allow visual inspection of big data on record level but focus on reporting and analytics (Bao and Chen, 2014; Kandel *et al.*, 2011; Liu *et al.*, 2013; Philip Chen and Zhang, 2014). Other solutions are available as native applications only (Childs *et al.*, 2011; Klinkert *et al.*, 2014). This framework represents a valuable new tool for visual inspection of conceptual chemical spaces as a help to cope with big data in chemistry for the case of large databases of molecules (Tetko *et al.*, 2016).

2 Materials and methods

2.1 System architecture

FUn implements a client-server model, communicating over two different channels. While web assets such as scripts, markup and images are loaded through HTTP, asynchronous communication upon user interaction is carried out using the WebSocket protocol. WebSocket was chosen to reduce wire-level and protocol overhead as well as for its support of potential future real-time data sources, given its full-duplex capabilities. The data consumed by the server application are pre-preprocessed using a flexible and automated Python-based toolchain carrying out PCA, 3-dimensional binning, indexing and the creation of color maps. Part of this toolchain is an HTTP-based web service allowing for the projection of additional data onto the visualized space. This toolchain was developed in-house and is publicly available. The client application Faerun consists of static HTML, CSS and JavaScript files, handling user input, communication and visualization. A more comprehensive description of the system architecture can be found on the project website (http://doc.gdb.tools/fun).

2.2 Client (Faerun)

Core of the client application is the JavaScript 3D data visualization engine Lore.js, which wraps the WebGL API and is optimized for rendering large scatter plots and graphs. We chose to implement Lore.js to optimize the performance of all aspects of the WebGL rendering pipeline as well as to control main memory and central processing unit (CPU) usage compared to THREE.js which was used in WebDrugCS and WebMolCS. SVG-based or 2D visualization libraries such as D3.js were not considered due to low performance and high memory usage when drawing more than $\sim 10^3$ data points or the inability to render 3-dimensional scenes (Bostock *et al.*, 2011). The main performance enhancing features encompass (i) Data points are stored in an octree, a space partitioning data structure, reducing the time complexity of operations such as vertex picking or k-nearest-neighbor searches. (ii) The sorting algorithm used throughout the engine is a radix sort implementation for 32-bit floats with a time complexity of $O(4n)$. (iii) All data are stored in typed arrays, reducing the memory usage for 32-bit integers and 32-bit floats by 50%. (iv) Instead of rendering spheres as separate geometries, or instances of a geometry, a custom-built fragment shader simulates spherical vertices using the sphere equation and a linear shadow function to enhance the perception of depth in the orthogonal projection. (v) Adaptive framerate. While the user is not interacting with the rendered scene (except for vertex picking), the framerate is lowered by 50–75%, reducing the number of context switches. We chose to implement features such as sorting, filtering and octree construction on the CPU, thus not impairing the rendering performance of the graphical processing unit dedicated to running our optimized shaders.

2.3 Server (Underdark)

The server application is written in the Go programming language and exposes a WebSocket based API using the Gorilla WebSocket implementation for Go (http://www.gorillatoolkit.org/pkg/websocket), allowing access to datasets stored on the filesystem of the server. Go drivers for a wide variety of databases are freely available (https://awesome-go.com/). To enhance loading times on slower connections, data are compressed by the server using DEFLATE.

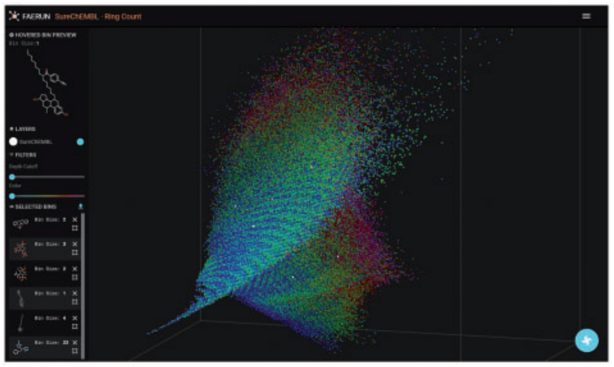

Fig. 1. Responsive graphical user interface. Faerun implements a responsive web interface based on the materialize.css (http://materializecss.com/) framework. The data are presented as an orthogonally projected 3D scatter plot. Multiple data points can be selected and summarizations of the contained data are shown to the left. Details on one or multiple selected data points can be shown in a new browser tab upon clicking a selected item

3 Results

As with our previously published applications, the SureChEMBL reference implementation provides access to multiple chemical spaces generated using different chemical fingerprints (Awale and Reymond, 2016; Awale *et al.*, 2017; Ruddigkeit *et al.*, 2013). The point coordinates are the result of a PCA dimensionality reduction on the high-dimensional fingerprint vectors, followed by a digitization into bins. Thus, one data point can contain multiple molecules. Depending on fingerprint and resolution, the number of data points rendered ranges from 0.5 to 8 million. Fingerprint features of interest are visualized by loading color maps, each representing a simple or composite feature. Molecular structures are drawn using SmilesDrawer (Fig. 1).

A set of tools is available to explore the visualized data. (i) Custom molecules (encoded as SMILES) can be projected onto the currently loaded space and used to filter the SureChEMBL data by applying the k-nearest-neighbors algorithm. (ii) A search function is available, allowing to search for a compound by its SureChEMBL id or SMILES string. (iii) The rendered data points can be filtered by color (representing a feature) or z-depth. (iv) Multiple data points can be selected for comparison. In case of the reference implementation, a data point can be expanded to show its content using a second instance of the Lore.js engine in a new browser tab. (v) Subsets of data, such as selections, bin contents or the result of applied filters, can be exported as csv files.

4 Conclusion

In summary, we believe that FUn, a framework facilitating easy web-based and interactive visualization of large, high-dimensional datasets, is a powerful tool to provide an additional route of access to big data, which can often only be searched or summarized. Browsing and exploring even vast amounts of data in an intuitive way can support and validate computational methods such as virtual screening through the integration of expert knowledge, especially when applying said methods to a novel data source. We see its application in both bioinformatics and cheminformatics, enabling easy to implement and versatile web-based access through visual inspection for very large and/or high-dimensional datasets found in genome-wide SNP data, microarray data, virtual screening results or MRI and microscopy voxel data as well as databases such as the Human Metabolome Database or the RCSB Protein Data Bank.

Funding

This work was supported by the Swiss National Science Foundation and NCCR TransCure.

Conflict of Interest: none declared.

References

Awale,M. *et al*. (2017) WebMolCS: a web-based interface for visualizing molecules in three-dimensional chemical spaces. *J. Chem. Inf. Model.*, **57**, 643–649.

Awale,M. and Reymond,J.-L. (2016) Web-based 3D-visualization of the DrugBank chemical space. *J. Cheminform.*, **8**, 25.

Bostock,M. *et al*. (2011) D3 Data-Driven Documents. *IEEE Trans. Vis. Comput. Graph.*, **17**, 2301–2309.

Childs,H. *et al*. (2011) VisIt: an end-user tool for visualization and analyzing very large data. In: *Proceedings of SciDAC 2011*, Denver, CO, USA.

Bao,F. and Chen,J. (2014) Visual framework for big data in d3.js. In: *2014 IEEE Workshop on Electronics, Computer and Applications*. IEEE, pp. 47–50.

Kandel,S. *et al*. (2011) Wrangler. In: *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*. ACM Press, New York, New York, USA, p. 3363.

Klinkert,I. *et al*. (2014) Methods for full resolution data exploration and visualization for large 2D and 3D mass spectrometry imaging datasets. *Int. J. Mass Spectrom.*, **362**, 40–47.

Liu,Z. *et al*. (2013) imMens: real-time visual querying of big data. *Comput. Graph. Forum.*, **32**, 421–430.

Papadatos,G. *et al*. (2016) SureChEMBL: a large-scale, chemically annotated patent document database. *Nucleic Acids Res.*, **44**, D1220–D1228.

Philip Chen,C.L. and Zhang,C.-Y. (2014) Data-intensive applications, challenges, techniques and technologies: a survey on Big Data. *Inf. Sci. (Ny)*, **275**, 314–347.

Ruddigkeit,L. *et al*. (2013) Visualization and virtual screening of the chemical universe database GDB-17. *J. Chem. Inf. Model.*, **53**, 56–65.

Tetko,I.V. *et al*. (2016) BIGCHEM: challenges and opportunities for Big Data analysis in chemistry. *Mol. Inform.*, **35**, 615–621.