


# Loch Prospector: Metadata Visualization for Lakes of Open Data

Neha Makhija 

Mansi Jain 

Nikolaos Tziavelis 

Laura Di Rocco 

Sara Di Bartolomeo 

Cody Dunne 

Northeastern University

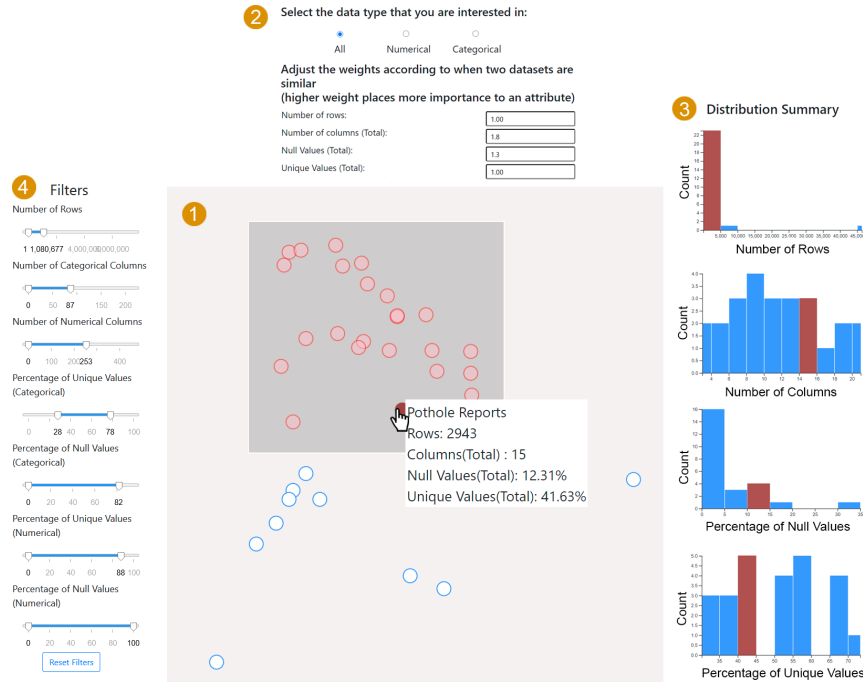


Figure 1: LOCH PROSPECTOR visualizes available datasets in Open Data lakes using four linked components. A multidimensional scaling (MDS) [16] plot ① shows a point for each dataset, organized spatially by similar metadata characteristics. Weights for the MDS algorithm can be tuned for particular types of metadata using the Visualization Configuration Box ②. Dynamic Filters [2] ④ can be used to explore datasets of interest, with Summary Statistics ③ shown for the currently selected datasets.

## ABSTRACT

Data lakes are an emerging storage paradigm that promotes data availability over integration. A prime example are repositories of Open Data which show great promise for transparent data science. Due to the lack of proper integration, Data Lakes may not have a common consistent schema and traditional data management techniques fall short with these repositories. Much recent research has tried to address the new challenges associated with these data lakes. Researchers in this area are mainly interested in the *structural* properties of the data for developing new algorithms, yet typical Open Data portals offer limited functionality in that respect and instead focus on data *semantics*. We propose LOCH PROSPECTOR, a visualization to assist data management researchers in exploring and understanding the most crucial structural aspects of Open Data — in particular,

metadata attributes — and the associated task abstraction for their work. Our visualization enables researchers to navigate the contents of data lakes effectively and easily accomplish what were previously laborious tasks. A copy of this paper with all supplemental material is available at [osf.io/zkxv9](https://osf.io/zkxv9)

**Index Terms:** Human-centered computing—Visualization

## 1 INTRODUCTION

Recently, the database community has shifted its attention to the data management challenges introduced by data lakes (e.g., [21] [26] [36]). In this paper, we focus on lakes of Open Data [11] [32] due to their prevalent use in data science [19] and by governments and organizations embracing data transparency. Data in these lakes is usually stored in a tabular format but is mainly semi-structured — often as CSV files — due to the dynamic nature of the dataset. Therefore, data in Open Data lakes may lack important *structural information* typically found in a traditional database management system such as column names, data types, and functional dependencies.

Before a researcher is able to develop, optimize, or test algorithms that operate on a lake of Open Data, they must first (1) gain insight into the variation in structural properties and (2) filter to an appropriate subset of the data lake. Understanding the structural properties of data in the lake is key for algorithm design, as these properties directly affect algorithmic operations and performance. E.g., the recommended algorithms for searching, cleaning, and pro-

\*Corresponding author. E-mails: [ makhija.n | jain.man | tziavelis.n | la.dirocco | dibartolomeo.s | c.dunne ]@northeastern.edu

filing a data lake are different for small vs. big tables [22] [27] [35]. However, *semantic differences* — e.g., whether the dataset is about agriculture or finance — typically are irrelevant for such algorithms. The ability to find real-world datasets that have particular shapes, value distributions, or other corner case structural characteristics can also facilitate the design of standardized benchmarks.

Currently, researchers invest a lot of time to accomplish the above tasks. They often must fish for the right datasets from a long list via keyword search or develop their own custom filtering tools. Existing portals for Open Data repositories incorporate search capabilities that display only some metadata such as the title or the date of publication. Although these are useful for understanding the domain (semantic) relevance of the dataset, the useful structural metadata for a researcher is often not provided or hard to find.

We aim to help data management researchers to more quickly and easily assess the structural aspects of lakes of Open Data. Following a design study “lite” methodology [30], we worked with domain experts over 7 months to assess their needs and design effective visualizations to assist them in their tasks.

Here we detail initial contributions from an ongoing design study:

- A *task abstraction* for a data management researcher interested in designing tools atop lakes of Open Data.
- The *design and implementation* of LOCH PROSPECTOR, an interactive visualization tool for exploring available datasets in a data lake based on their structural metadata. LOCH PROSPECTOR is web-based and open-source.
- *Initial validation* of our system design with a usability study and expert feedback.

A copy of this paper, source code, and data are available at [osf.io/zkxv9](https://osf.io/zkxv9) and a demo is online at [lochprospector.github.io](https://lochprospector.github.io)

## 2 RELATED WORK

Data lakes are a recent trend, hence there are few visualizations designed to be used in concert with them. E.g., Krause [15] describes the construction of a data lake and an accompanying visualization specifically for the health care domain. However, visualizing Big Data, often the content of a data lake, is well-studied [3] with a focus on issues related to scalability [10] [20], diversity [12], or high-dimensionality [18]. We address a different set of challenges around visualizing the structural properties of the data.

Many interactive visualizations have been created to provide better *semantic metadata* query interfaces than classic text-based and rank-list approaches. Such visualizations are most useful when the search query or user intent is vague or ill-defined, as in our case. Here we list only a few relevant examples. FacetMap [29] focuses on facilitating searching and browsing in personal information stores by using the associated semantic metadata for filtering. VisMeB [14] also visualizes semantic metadata by combining table views with other interactively linked idioms such as scatterplots. For web-based information retrieval, Lighthouse [17] uses a hybrid ranked-list and document-clustering approach. WebSearchViz [23] similarly uses visual encodings to represent the semantic relationships between web-search queries and relevant Web pages. No previous approach we are aware of focuses on visualizing *structural metadata*, but we can draw inspiration from these existing designs.

One key inspiration from previous work is our choice of using spatial layout to encode similarity among search results. Unlike those designs [14] [17] [23] [29], our approach relies on multidimensional scaling (MDS) [7] [16]. MDS transforms a set of data into coordinates (typically 2-dimensional) by optimizing the layout to visually encode a given similarity function with the spatial layout. MDS has been widely used as a dimensionality reduction method for visualization research, e.g., [24]. It has also been used in visualizations with complex data types, including temporal data [4]. Here, we use MDS to encode weighted relationships between datasets based on their *structural* properties — not their *semantic* properties.

## 3 PROBLEM DOMAIN

LOCH PROSPECTOR was developed in a classroom setting following the the design study “lite” methodology [30] — which is designed specifically for teaching and learning settings. We collaborated with the DATA Lab at Northeastern University [1], a research lab in our institution which focuses primarily on issues related to efficient data management and querying. Dr. Laura Di Rocco, a postdoctoral researcher working on multiple such projects was our main test case to help us understand the needs of Open Data researchers. Her invaluable contributions led to including her as an author here.

### 3.1 Domain Challenges

We conducted interviews to identify the tasks of this data management researcher that would most benefit from an appropriate visualization, and hopefully would benefit other researchers as well. Researchers on data management and querying, like most researchers, have many duties. These range from the identification of open problems to the development and testing of new algorithms. To guide the whole process, a precise understanding of the structural properties of available data is considered invaluable. Ideally, a researcher would like to have quick answers to questions such as, “Are the tables mostly short and wide or tall and thin?” or “For the small tables, are there a lot of missing values?” Additionally, working with a small subset of the data is necessary for rapid prototyping of new solutions. Yet, finding the right subset is deceptively difficult since searching the repositories according to the structural metadata of interest requires much manual effort. As a result, researchers often sacrifice representativity and settle for artificial examples from synthetic data generators. Another major limitation of current research is the absence of rigorous evaluation methods. Finding a proper testing subset from Open Data repositories could also help towards that direction, increasing the transparency of the evaluation.

### 3.2 Task Analysis & Abstraction

Our interview notes exposed key iterative tasks in the workflow of a data management researcher that may benefit from visualization. We then abstracted the tasks using Brehmer & Munzner’s typology [6].

[IDENTIFY] The first domain task is to find datasets that meet the user’s criteria based on statistical aspects extracted from the metadata. These criteria are a combination of priorities based on dataset size, number of numerical/categorical columns, etc. At the highest level, this is a *discover* task where the user wishes to find appropriate dataset(s). This is an *explore* task at the mid level since the ideal dataset(s) and where they can be found is unknown. The lowest-level action is to *identify* the dataset that matches the user’s need by selecting from the visualization.

[SUMMARIZE] The second task is to summarize a specified set of datasets based on overall structural information, i.e., statistical measures of the metadata distribution. [SUMMARIZE] can apply to all datasets or a selected subset. As before, the highest-level action is to *discover* valuable information about our datasets. Likewise, it is an exploration task since the user is not looking for a specific piece of information and the set is not predefined. However, this task is at the lowest level a typical *summarize* task.

These general tasks should apply to other researchers using data lakes who wish to find appropriate datasets for their research, gain insights about the data, build synthetic benchmarks, or to otherwise verify the results of the algorithms they develop.

### 3.3 Data

In this initial study we use data provided by the Open Data portal of the U.S. government [32], which consists of hundreds of thousands of tables with high heterogeneity. We randomly sampled 200 tables, accessed their contents, and compute and stored structural metadata. Fetching and pre-processing of the data was done in Python — `data/download_data.py` in the supplemental material

at [osf.io/zkxv9](https://osf.io/zkxv9). For each table, we calculated a relevant subset of structural properties. We counted the number of rows, columns, categorical and numerical columns, unique values, and null values. We also calculated the percentage of unique and null numeric and categorical values. These values were extracted and stored in a single CSV file — `data/final_metadata.csv` in the supplemental material. Although this paper focuses on Open Data from the United States government, our tool can be used for any data lake as it is agnostic to the data semantics.

## 4 LOCH PROSPECTOR VISUALIZATION

We designed an interactive visualization tool — LOCH PROSPECTOR — to help data management researchers to explore and understand the datasets available in lakes of Open Data. LOCH PROSPECTOR is web-based and open-source. A demo is online at [lochprospector.github.io](https://lochprospector.github.io) while code and data are available at [osf.io/zkxv9](https://osf.io/zkxv9). Our designs are based on our task abstractions and compiled *structural* metadata.

Following the Shneiderman’s Mantra [28] of “overview first, zoom and filter, and details on demand,” the visualization initially shows all available datasets in the data lake. Fig. 1 shows all the visualization components, which are described in detail in the following sections and are joined together as multiple coordinated views [25] [33]. A multidimensional scaling (MDS) [16] plot ❶ shows an overview with a point for each dataset, grouping datasets with comparable structural properties together. Different structural properties can be weighted separately and used as input to the MDS algorithm by editing the Visualization Configuration Box ❷. The Summary Statistics ❸ panel shows an overview distribution of values for each of these structural properties. Finally, Dynamic Filters [2] ❹ and brushing the MDS plot ❶ allow the user to filter to relevant subsets. The Summary Statistics ❸ panel updates to show structural property distributions for only these filtered datasets. This visualization model allows the user to quickly switch between these two tasks; i.e., to iterate between [IDENTIFY] and [SUMMARIZE] tasks to refine the subset of relevant datasets.

### 4.1 Development Process

We made several iterations of design sketches for the visualization, experimenting with different visual encodings, channels, linkings and views to represent our data effectively. We walked through how a user would execute their tasks for each of these sketches and routinely obtained feedback from the data management researcher to ensure that our sketches were in line with their priorities.

Our initial design sketches consisted of using a combination of position, shape, and color visual encodings to explicitly display all the metadata attributes that the user cares about. However, we ran into difficulties with using separate encodings for the many structural properties a researcher could be interested in. Additionally, we realized that while a fixed per-attribute encoding might suffice for certain use cases, we could strive for a more general solution in which the visualization adapts when different attributes hold different levels of importance for different use cases. This could be done using Multidimensional Scaling (Section 4.2) that allows users to weigh how important each attribute is to them. Thus, we created a visualization that shows the overall relationships between datasets based on an arbitrary number of attributes.

### 4.2 Multidimensional Scaling Plot and Weighting

The main focus of the visualization is a multidimensional scaling (MDS) plot [7] [16]. As shown in Fig. 1 ❶, Each dataset is shown by a distinct disk in the plot, such that “similar” datasets are generally placed closer to each other than other pairs that are less similar. The names of each dataset and an option to download them are provided on demand, i.e., by hovering over a glyph that represents the dataset and by clicking. This MDS plot supports the [IDENTIFY] task.

MDS transforms a set of data with a similarity function into a set of coordinates (typically 2-dimensional), where points closer together tend to indicate a higher similarity than points far apart. This dimensionality reduction approach substantially reduces the number of visual encodings we would have required. In line with Tufte’s recommendations of graphical integrity [31], the resulting visualization has a much better data-to-ink ratio as well — albeit at the cost of losing some information in the process.

Computing an MDS layout requires first defining a similarity measure. Here we use a Euclidean distance function that takes into account user-defined weights to alter the importance of the specific structural metadata attributes under consideration. By default, all available properties are weighted equally.

The similarity function is currently defined using the following four properties that we chose according to input from the data management researcher we interviewed:

1. *The number of rows* is a common metadata attribute that influences algorithmic performance, in particular scalability.
2. *The number of columns* is likewise important, though can be transposed with rows.
3. *The percentage of null values* is often an indicator of dataset quality or completeness. Datasets with few null values are particularly useful for ML training tasks, while datasets with many null values are good for developing approximate solutions as they are easier to store.
4. *The percentage of unique values*, i.e., values that appear only once in the corresponding column. Unique values is an important measure for detecting primary keys, correlations, and functional dependencies.

Using this similarity function, we generate a 2D position for each dataset using an open-source JavaScript MDS implementation [9].

The user has the ability to control the MDS Plot by appropriately modifying the weights of the similarity function using the Visualization Configuration Box (Fig. 1 ❷). This allows users to prioritize the properties most relevant for their use case. For instance, if the user is looking for datasets according to their number of rows, but the percentage of null values within the dataset makes no difference to them, they may set the weight of the number of rows attribute to 10 and the weight of the null percentage attribute to 0.

The Visualization Configuration Box (Fig. 1 ❷) also allows allows users to choose between three modes: whether they want to include categorical or numerical dataset columns, or both. This is implemented by radio buttons, and clicking on a radio button changes the attributes that are visible on screen.

### 4.3 Summary Statistics

In conjunction with the centered plot, we use histograms to show the distribution and range of values for the structural properties we are interested in across datasets. These are the number of rows, number of columns, percentage of unique values, and percentage of null values. These histograms are demonstrated in Fig. 1 ❸. Showing these distributions and ranges helps to satisfy the [SUMMARIZE] task and informs iterative refinement of the query set by the user as they move between the [IDENTIFY] and [SUMMARIZE] tasks.

### 4.4 Dynamic Filters

LOCH PROSPECTOR also includes a set of dynamic filters [2] to assist in the [IDENTIFY] task. These are implemented as double-ended sliders with the data extent as their maximum range, visible in Fig. 1 ❹. These filters allow users to specify any mandatory requirements or to quickly eliminate outlier ranges from the data.

### 4.5 Linking Views

To support query refinement by iterating between the [IDENTIFY] and [SUMMARIZE] tasks, we use multiple coordinated views [25] [33] connected through interactivity.

The MDS plot and histograms are linked such that brushing or mousing over a set of points in the MDS plot immediately displays their property distributions in the histograms. This interaction is illustrated in Fig. 1 for the “Pothole Reports” dataset, highlighted in dark red. Conversely, hovering over bars in the histograms highlights the corresponding points in the MDS plot. Users may use this interactivity to iteratively filter to datasets with property distributions of interest or to discover summary statistics about a given set of points.

Individual datasets in the MDS plot can also be hovered over to reveal more details such as the name of the dataset and its structural properties. This interaction is also visible in Fig. 1. Datasets can also be clicked to open download links for them.

Naturally, interacting with the dynamic filters updates the set of included data points. Fig. 1 ④ shows most of the range filters being used for this example. Likewise, the filter radio buttons in ② can be selected to update the data types of interest.

Finally, the weights of the MDS similarity function can be modified to refine the MDS plot in real-time. Fig. 1 ② shows a set of weights prioritizing the number of columns heavily and, to a lesser degree, the percentage of null values. This would help a researcher that, e.g., is searching for relatively complete and large datasets.

## 4.6 Overall Design

LOCH PROSPECTOR uses a single-page design with multiple juxtaposed visualization and interaction widgets (Fig. 1). We chose a juxtaposed layout so users can quickly and easily explore multiple data dimensions without needing a complicated superimposed or composite design [13]. Our single-page design also reduces the need to remember previous aspects of the data while exploring another, which would suffer if users had to scroll or otherwise change views. Users are also likely to interact more rapidly when fewer interactions are required to explore multiple views. The layout of the visualization was designed to keep the MDS plot in the center with the filters, distribution charts, and additional interactive elements alongside.

## 5 EVALUATION & DISCUSSION

### 5.1 Usability Testing

Before introducing LOCH PROSPECTOR to our collaborators in the data management research group, we undertook informal qualitative usability testing to validate our implementation and the basic elements of our design. This is in line with the suggested validation in the design study “lite” methodology [30]. 20 Computer Science graduate students at our institution participated in the study.

The feedback reassured us that the layout was clear and the interactions were logical and intuitive. The usability study also helped us with bug-fixes and provided ideas for feature enhancements. At the same time, we realized that the plot created by MDS is not very intuitive for first-time users and it requires some explanation beforehand. Part of this confusion is due to the fact that coordinates are determined by the algorithm and not standard axes as in scatterplots. Thus, without proper explanation it is unclear to an unfamiliar user what the axes mean or how the data points have been clustered. To prevent this confusion, we preface the visualization with a short explanation of MDS and how the similarity function is calculated.

### 5.2 Expert Feedback

To validate our system, design, and abstractions from a domain perspective we introduced LOCH PROSPECTOR to the data management researcher we originally interviewed. As part of an in-person feedback session, they used the visualization to find data with a specific set of properties relevant to an algorithm they were designing. They found our interface to be quite useful, and said that without the tool, they would accomplish the same task using custom code to profile and subset the data. This would have required more time and has a much higher barrier to entry as there is no simple visual interface. Custom code also requires users to pre-specify their requirements.

This is inconvenient when the requirements are complicated, not set in stone, and different across different tasks. With LOCH PROSPECTOR, they were able to perform that same task in about 10 minutes — while interactively specifying and modifying their requirements based on the data distribution shown on screen.

The data management researcher said that the interface is simple to use, with no particularly confusing elements — aside from the axes in the MDS plot. This reinforced our earlier feedback and pushed us to add further clarification. Thus we added one-line summary of how the plot should be read directly beneath it.

We observed, and the researcher confirmed, the iterative nature of their work in which they would move back and forth between views for the [IDENTIFY] and [SUMMARIZE] tasks. This reassured us that our abstractions and design are on the right track to be useful for them and, hopefully, other data management researchers as well.

## 5.3 Future Work

Our visualizations from this in-process design study have several limitations. LOCH PROSPECTOR is implemented in JavaScript using D3 [5] and is currently limited to about 200 datasets with sub-100ms interaction response time. Since we use a classic MDS algorithm with cubic complexity, also in JavaScript, considering more than a few hundreds of datasets leads to substantial interaction delays. However, the exact results of MDS are not crucial, hence an approximation algorithm could be considered to lower the computational complexity and allow for high scalability. Multiple such approximation schemes could be used with LOCH PROSPECTOR, e.g., [8] [34]. Alternatively, server-side computation would reduce MDS delays for many datasets. Additionally, there are challenges related to visual design when the number of datasets significantly grows (e.g., occlusion and point picking). These may be alleviated through the use of transparency, lens-based selection techniques, or a further detail view for showing individual datasets in a range.

Apart from the [IDENTIFY] and [SUMMARIZE] tasks considered in Section 3.2, we have also identified *one more potential task* while talking to our partner — it would be useful for a researcher to [COMPARE] two specific datasets to find out whether they are unionable, i.e., if they have the same distribution of values. We decided not to address this task at the current stage of the project as it did not fit within the same iterative workflow LOCH PROSPECTOR is designed to support. To support a [COMPARE] task we could, e.g., juxtapose or superimpose the attribute distribution histograms for two datasets. If they are similar, the datasets may have been drawn from the same distribution and the tables could be unioned. This would be particularly useful for qualitatively validating the results of table union search algorithms [22]. The visualization can also be extended to accommodate users who care about both structural and semantic properties by integrating preexisting keyword filtering and text-based search options alongside the Dynamic Filters ④.

## 6 CONCLUSION

Motivated by the needs of data management researchers working on lakes of Open Data, we developed LOCH PROSPECTOR, a visual tool for finding datasets with the right set of structural properties. Our design is based on interviews with a data management researcher, from which we built a task analysis and abstraction. The resulting visualization was evaluated using informal usability testing as well as domain expert feedback. Expert feedback indicates that our abstractions are appropriate and that LOCH PROSPECTOR would transform a previously long and complex task requiring programming knowledge into a simple and fast visual exercise.

## 7 ACKNOWLEDGEMENTS

We thank the students of CS 7250 for testing LOCH PROSPECTOR and providing feedback and our Service-Learning TA Gayathri Raj.

## REFERENCES

- [1] DATA lab at Northeastern. <https://db.khoury.northeastern.edu/>.
- [2] C. Ahlberg and B. Shneiderman. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, p. 313–317, 1994. doi: 10.1145/191666.191775
- [3] G. Andrienko, N. Andrienko, S. Drucker, J.-D. Fekete, D. Fisher, S. Idreos, T. Kraska, G. Li, K.-L. Ma, J. Mackinlay, et al. Big data visualization and analytics: Future research challenges and emerging applications. In *BigVis 2020: Big Data Visual Exploration and Analytics*, 2020.
- [4] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics*, 22(12):559–568, 2015. doi: 10.1109/TVCG.2015.2467851
- [5] M. Bostock, V. Ogievetsky, and J. Heer. D<sup>3</sup>: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011. doi: 10.1109/TVCG.2011.185
- [6] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, 2013. doi: 10.1109/TVCG.2013.124
- [7] M. A. Cox and T. F. Cox. Multidimensional scaling. In *Handbook of data visualization*, pp. 315–347. Springer, 2008. doi: 10.1007/978-3-540-33037-0\_14
- [8] C. Faloutsos and K.-I. Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proc. 1995 ACM SIGMOD international conference on Management of data*, pp. 163–174, 1995. doi: 10.1145/223784.223812
- [9] B. Frederickson. mds.js. <https://github.com/benfred/mds.js>, 2015.
- [10] P. Godfrey, J. Gryz, and P. Lasek. Interactive visualization of large data sets. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2142–2157, 2016. doi: 10.1109/TKDE.2016.2557324
- [11] Government of Canada. Open data. <https://open.canada.ca/en/open-data>.
- [12] R. Hai, S. Geisler, and C. Quix. Constance: An intelligent data lake system. In *Proc. 2016 International Conference on Management of Data, SIGMOD '16*, p. 2097–2100, 2016. doi: 10.1145/2882903.2899389
- [13] W. Javed and N. Elmqvist. Exploring the design space of composite visualization. In *2012 IEEE Pacific Visualization Symposium*, pp. 1–8, 2012. doi: 10.1109/PacificVis.2012.6183556
- [14] P. Klein, H. Reiterer, F. Muller, and T. Limbach. Metadata visualisation with VisMeB. In *Proc. Seventh International Conference on Information Visualization, 2003. IV 2003.*, pp. 600–605, 2003. doi: 10.1109/IV.2003.1218047
- [15] D. D. Krause. Data lakes and data visualization: an innovative approach to address the challenges of access to health care in mississippi. *Online journal of public health informatics*, 7(3), 2015. doi: 10.5210/ojphi.v7i3.6047
- [16] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964. doi: 10.1007/BF02289565
- [17] A. Leuski and J. Allan. Lighthouse: Showing the way to relevant information. In *IEEE Symposium on Information Visualization 2000, INFOVIS 2000*, pp. 125–129. IEEE, 2000. doi: 10.1109/INFVIS.2000.885099
- [18] S. Liu, D. Maljovec, B. Wang, P.-T. Bremer, and V. Pascucci. Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268, 2016.
- [19] R. J. Miller. Open data integration. *Proc. VLDB Endowment*, 11(12):2130–2139, 2018. doi: 10.14778/3229863.3240491
- [20] D. Moritz, D. Fisher, B. Ding, and C. Wang. Trust, but verify: Optimistic visualizations of approximate queries for exploring big data. In *Proc. 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, p. 2904–2915, 2017. doi: 10.1145/3025453.3025456
- [21] F. Nargesian, E. Zhu, R. J. Miller, K. Q. Pu, and P. C. Arocena. Data lake management: challenges and opportunities. *Proc. VLDB Endowment*, 12(12):1986–1989, 2019. doi: 10.14778/3352063.3352116
- [22] F. Nargesian, E. Zhu, K. Q. Pu, and R. J. Miller. Table union search on open data. *Proc. VLDB Endowment*, 11(7):813–825, 2018. doi: 10.14778/3192965.3192973
- [23] T. Nguyen and J. Zhang. A novel visualization model for web search results. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):981–988, 2006. doi: 10.1109/TVCG.2006.111
- [24] A. Nocaj and U. Brandes. Organizing search results with a reference map. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2546–2555, 2012. doi: 10.1109/TVCG.2012.250
- [25] C. North and B. Shneiderman. Snap-together visualization: A user interface for coordinating visualizations via relational schemata. In *Proc. Working Conference on Advanced Visual Interfaces*, AVI '00, p. 128–135, 2000. doi: 10.1145/345513.345282
- [26] N. Noy. When the web is your data lake: Creating a search engine for datasets on the web. In *Proc. 2020 ACM SIGMOD International Conference on Management of Data*, p. 801, 2020. doi: 10.1145/3318464.3393815
- [27] M. Ota, H. Müller, J. Freire, and D. Srivastava. Data-driven domain discovery for structured datasets. *Proc. VLDB Endowment*, 13(7):953–967, 2020. doi: 10.14778/3384345.3384346
- [28] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proc. 1996 IEEE symposium on visual languages*, pp. 336–343, 1996. doi: 10.1109/VL.1996.545307
- [29] G. Smith, M. Czerwinski, B. Meyers, D. Robbins, G. Robertson, and D. S. Tan. FacetMap: A scalable search and browse visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):797–804, 2006. doi: 10.1109/TVCG.2006.142
- [30] U. H. Syeda, P. Murali, L. Roe, B. Berkey, and M. A. Borkin. Design study “lite” methodology: Expediting design studies and enabling the synergy of visualization pedagogy and social good. In *Proc. 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, p. 1–13, 2020. doi: 10.1145/3313831.3376829
- [31] E. R. Tufte. *The visual display of quantitative information*, vol. 2. Graphics press Cheshire, CT, 2001.
- [32] United States Government. Data.gov. <https://www.data.gov/>.
- [33] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proc. Working Conference on Advanced Visual Interfaces*, AVI '00, p. 110–119, 2000. doi: 10.1145/345513.345271
- [34] T. Yang, J. Liu, L. McMillan, and W. Wang. A fast approximation to multidimensional scaling. In *IEEE workshop on computation intensive methods for computer vision*, 2006.
- [35] D. Zhang, Y. Suhara, J. Li, M. Hulsebos, Ç. Demiralp, and W.-C. Tan. Sato: Contextual semantic type detection in tables. *arXiv preprint arXiv:1911.06311*, 2019.
- [36] Y. Zhang and Z. G. Ives. Finding related tables in data lakes for interactive data science. In *Proc. 2020 ACM SIGMOD International Conference on Management of Data*, pp. 1951–1966, 2020. doi: 10.1145/3318464.3389726