Visualizing and Understanding Dataset Search Results

Sonia Castelo*
scq202@nyu.edu
New York University

Erin McGowan*
erin.mcgowan@nyu.edu
New York University

Shirley Berry*
slb502@nyu.edu
New York University

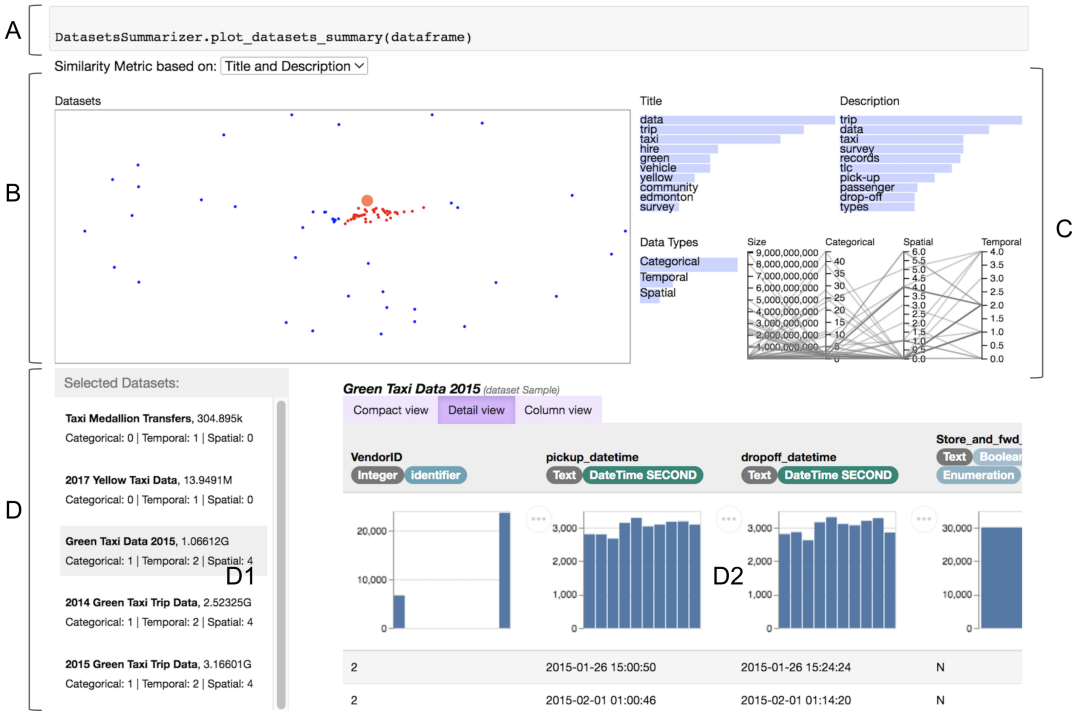

Figure 1: *DatasetsSummarizer* applied to compare the dataset search results for the keyword query "Taxi". A) The system is integrated with Jupyter Notebook and can be invoked with one line of code. (B) Datasets Similarity View: the data is displayed as a collection of points in a two-dimensional scatter plot where each point represents a dataset. (C) Datasets Summary View: exploring datasets through visualization of frequency of words, datasets statistics, and a parallel coordinate. (D) Datasets Presentation and Exploration View: the system displays the list of selected datasets (D1), and allows users to inspect the contents of each dataset, including column statistics and data types (D2).

ABSTRACT

The easy availability of dataset aggregators and search engines has made it easier than ever to get access to datasets on the web. However, the number of datasets available today has made it proportionally more challenging to parse through those results in an efficient manner to get the information you need to make an informed choice about those search results. We present the *DatasetsSummarizer* as a tool to help meet these challenges. In this paper we describe the problem we have set out to solve, the overall system and methods we used to create this solution, and a case study demonstrating how our tool can help make parsing dataset search results easier. We also discuss the limitations of our design and future work we hope will make this an even more powerful tool in the hands of users.

KEYWORDS

dataset search, data discovery, data visualization, dataset similarity, metadata

ACM Reference Format:
Sonia Castelo, Erin McGowan, and Shirley Berry. 2023. Visualizing and Understanding Dataset Search Results. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*All authors contributed equally to this research.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The process of finding similar datasets is challenging and time consuming despite being a crucial task in many fields of data science. Even when datasets are aggregated in tools like Kaggle [2] and Google Dataset Search [6], manually combing through the results is tedious. Due to the explosion in the number of datasets available in such search repositories, it is easier than ever to discover datasets, but increasingly difficult to sift through the noise of search results to find relevant, high-quality data. Finding multiple datasets that can be used in conjunction with one another is even more challenging.

Although many dataset search tools provide a summary and sample of each dataset, it is difficult to tell at a glance whether your search has yielded any useful results, and whether or not the datasets in the result are good candidates to be joined or used together. There are tools available on the market that are capable of comparing datasets, but most of them are expensive and intended for use at an enterprise level [1] or are a tool primarily for another purpose that requires the user to manually apply it to the datasets they want to compare [13]. For users of open datasets, this presents an insurmountable barrier to entry. In addition, most dataset search tools do not include this comparison capability. Auctus [9] is the exception to this, providing a way to find similar and joinable datasets for a selected dataset. While this tool works well if you have a dataset to start with, if you do not have that first target dataset you must compare the datasets in your search one by one, a total of $\binom{n}{2}$ manual comparisons. The task only becomes more more arduous if you want to find and use more than two datasets. It also does not provide users an easy way to see data about the search result as a whole.

We address this problem through the following contributions:

- We create an interactive visualization that describes and compares features of a large number of datasets in a single view.
- We construct the architecture of the data preprocessing pipeline used to create this visualization in a manner that lends itself to integration with existing dataset search engines.
- We demonstrate the effectiveness of this tool by presenting a use case wherein we compare the dataset discovery experience while using our tool with the experience of using an existing dataset search engine, Auctus.

2 PROBLEM FORMULATION

Dataset search is still a largely manual, human-driven, and yet data intensive process. It is frequently iterative and involves navigating between multiple datasets in order to accomplish the task, whether that task is exploration or searching for data to answer a very specific question. A frequent activity in data exploration is finding datasets that can work well together to accomplish a goal, something that one study calls "linking"[12]. They further described this task as "finding commonalities and differences between two or more datasets". These similarity comparisons are not only challenging to conduct manually but can also be computationally expensive, especially when applied to a large number of comparable items.

This study also identified a behavior among those they surveyed of starting their search with a broad categorical keyword, and manually narrowing the results down[12]. Although their survey was conducted with a limited number of participants (n = 20), they found this observation to be backed up statistically, with the average search length in the logs they analyzed being 2.44 words. From there, users would look through the datasets manually to see if any of them matched the more specific criteria they had in mind. This highlights the need for any solution to be user-centric and driven by the fact that finding datasets is often an open-ended, exploratory journey, rather than something specific like searching for an article in Google.

While fully comparing the contents of two datasets would give the most accurate and complete metric of their similarity to one another, doing so would be prohibitively expensive given the number of datasets available in these repositories and the size of the datasets themselves. In addition, being able to interact with and explore these similarity measurements is key if the goal is to provide users a way to move through the datasets in their result. Therefore, speed and responsiveness are an important feature, and any solution will require some tradeoffs between accuracy and efficiency. Consideration also needs to be given to what features are useful factors in calculation similarity, and what features about datasets are the most useful and informative when evaluating a search result.

3 RELATED WORKS

As far as we are aware, no dataset search engine currently has a feature that allows the user to visualize and compare the characteristics of datasets returned by a search. We believe this is due to several more general challenges associated with visualization of big data. Many of these challenges were outlined by Rajeev Agrawal et al. [4], who cite the characteristics described in Gartner's 3Vs definition of "big data" (high volume, high velocity, and high variety) as the foundation of big data visualization obstacles. They explain that many datasets are too large to fit in memory and may be distributed across a cluster. Moreover, while some datasets are structured, organized, and stored in a traditional relational database format, others are large volumes of unorganized and unstructured data. Once you are able to find an efficient way to process such data, you will also find your ability to visualize it is limited both by human perception (human eyes generally struggle to extract meaningful information from a visualization once the data becomes too large) and the limited size of your screen.

In such cases, traditional data visualization methods will not suffice. In our case, the large volume of large datasets returned by a search query as well as the high variety between and within those datasets makes it difficult to visualize and compare them. Rajeev Agrawal et al. recommend two solutions for tackling these issues: 1) data reduction (namely sampling, filtering, and binning) and 2) reducing latency (pre-computing data when possible, parallelizing data processing and rendering, including a predictive middleware). We intend to take advantage of pre-computation in our processing pipeline to support both scalability as well as interactivity in the tool itself.

Although comparing a large number of datasets is not currently a feature of the major dataset search engines, there has been some related work on dataset comparison. For instance, VennPlex [8] allows users to visualize and compare features of large, high-variety

datasets using venn diagrams. However VennPlex differs from our system in three major respects: 1) it can only compare up to four datasets, 2) it is not interactive, and 3) it is heavily tailored to the characteristics of large and complex genomic and proteomic datasets.

Google Dataset Search implements a system for comparing datasets but this system is designed to identify and remove duplicates and replicas [6] and as such doesn't meet the needs of a user trying to analyze the results of a search. They base their comparison on metadata about the search results like title and description, as we have also done, as well as an available schema.org tag [3] to indicate that two URLs represent the same entity. They also analyze the datasets based on domain and download URL because these factors are good indicators of a duplicated entry. Because their system is intended as an automated way to detect duplicated and replicated results, the features that they use for this calculation aren't as applicable to a user trying to make sense of a list of datasets and how they might be related to one another. Furthermore, they don't present any kind of interactive visualization of these results, as we have.

We intend to compare our interactive visualization for dataset discovery to the current method of dataset discovery available on Auctus [9]. Auctus is a dataset search engine which allows users to find datasets via keyword queries and various filters. It also includes visualizations of certain features for each dataset, which allows users to manually compare their search results. However, this comparison can only be done by clicking through each result and viewing their respective visualizations one at a time. Auctus does not currently contain a way of comparing all of the datasets returned by a search query at the same time.

4 METHODS

We have created an interactive visualization called *DatasetsSummarizer* that describes the features of a large set of datasets (such as the results returned by a keyword search in a dataset search engine) and allows users to compare them in a single view. This tool gives you metadata information about all of the datasets returned from your search in an easy-to-understand visual format. In one glance, a user can identify interesting features about their search results like words that appear frequently, and characteristics about all of the datasets like what types of columns they have (see Figure 1).

4.1 Architecture

The high-level architecture of our system is depicted in Figure 2. In what follows, we describe its key components.

4.2 Data Ingestion

Auctus is the source of the data that we are using to conduct our similarity calculations. The first entry point into the program, as can be seen in the full pipeline notebook, is to conduct a keyword search against the Auctus REST API [9]. For the purpose of iterating on and testing this feature, we used the set of datasets returned for a keyword search on "taxi". A hard-copy of the dataset IDs that were returned by this search as of the date of publication can be found here.

The notebook collects all of the returned results, each of which contains a unique ID generated by Auctus and metadata which was

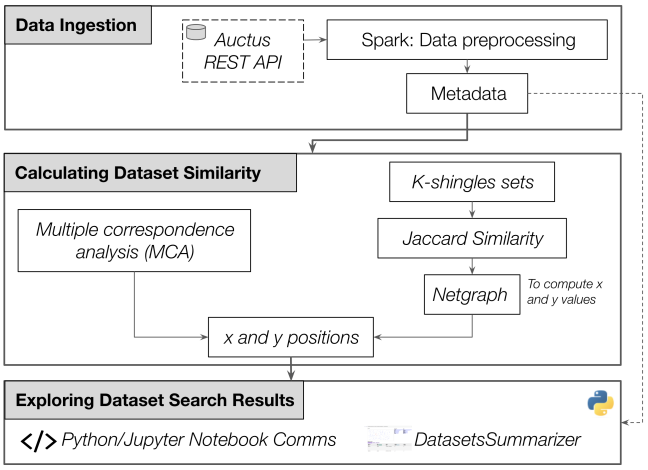

Figure 2: Overview of *DatasetsSummarizer*' architecture.

pre-generated for those datasets using Auctus's datamart-profiler. This metadata includes a title and description, which we use as part of our set of similarity calculations. We remove any non-significant words from the title and description, like "data" and "the". It also includes a list of the dataset's columns and their calculated semantic types. Auctus determines these semantic types based on a preprocessed analysis of the columns' contents and conforms to the standards set out by schema.org. These semantic types are used to provide combinations of column titles that are used in an additional similarity measurement. The columns whose semantic type is categorical, temporal, or spatial are first normalized and then gathered into an array. The normalization removes non-ascii characters from the names, replaces whitespace with underscores, and converts all of the column titles to lowercase. These metadata results are then saved into a Spark dataframe for use in the calculations.

Using the Auctus search API presents a reproducibility challenge due to the unavoidable volatility in open datasets. In these dataset aggregation systems a keyword search is non-deterministic in its results: new datasets might be added or removed, causing an identical search to return different results depending on the date they are conducted. The datasets themselves might change, for example through the removal of a column, causing their metadata to change. A dataset's description might be updated or even deleted.

In addition, because the primary goal of this tool is to present comparisons across a large number of datasets, saving hard copies of these datasets is impractical and wouldn't actually persist all of the data that we need. Even if we had saved copies of the datasets themselves and run the datamart-profiler on them to replicate what we would get back from Auctus, we would not get information like the dataset's title and description that isn't contained in the dataset itself.

Instead, to support replicating these results, the notebooks write the intermediate metadata results and the list of IDs that were retrieved to CSVs. Both filenames and the metadata itself include the date. In addition, the list of dataset IDs includes the "Last Modified" date from Auctus. To reproduce the visualizations referenced in

this paper, the taxi metadata CSV can be run through the start from metadata notebook.

4.3 Calculating Dataset Similarity

In order to visualize a meaningful comparison of a set of datasets, we must compute the similarity between those datasets with respect to a list of attributes (e.g. any combination of title, description, column names, etc.). We use the Spark dataframe output by the process described in the previous subsection to obtain these attributes for each dataset returned by a search query in Auctus. For each of these attributes, we create a set of k-shingles for each dataset and compute the jaccard similarity between each pair of k-shingle sets. The default value of k is five for relatively shorter attributes (e.g. title only) and nine for relatively longer attributes (e.g. description). If multiple attributes are specified, they are concatenated into a single string for each dataset before jaccard similarity is computed.

The final visualization contains clusters of dots, each representing one dataset, with the distance between each pair of dots representing how similar they are. Intuitively, dots should be closer together if they are more similar and father apart if they are less similar, and therefore must compute the jaccard distance using the output of our jaccard similarity calculations. We subsequently normalize these jaccard distances between the values 0.0001 and 1.0001. This will ensure that the matrices will not break the constraints of the least squares optimization we will perform using these values later (specifically, it will ensure that the matrices do not have a determinant of zero).

We then use these distance measures to compute the x and y position of each dataset in the multidimensional projection component (see section 3.3). We use the Netgraph [7] get_geometric_layout function to calculate the x and y coordinates of each dataset in the cluster based on the edge lengths (jaccard distances) between them. This function treats the cluster like a force directed graph and uses a sequential least squares programming optimizer to converge on a final layout. The nodes in this graph are datasets, and the length of an edge between two dataset nodes is equal to the jaccard distance between the datasets. These edges will not be visible in the final visualization.

We also utilize multiple correspondence analysis (MCA) to compute the similarity between the column names within each pair of datasets. Since MCA requires an input that is sorted into multiple categories, we will sort these column names by whether they represent a "categorical," "spatial," "temporal," or "other" type of variable. Earlier we used the Netgraph get_geometric_layout function to find the x and y position of each dataset within each similarity cluster. This function normalizes these positions between 0.05 and 0.95 so that the dots are framed nicely within the 1 by 1 background of the cluster visualization. We therefore normalize the x and y position of each dataset within the column name similarity cluster between 0.05 and 0.95 to match the netgraph padding. These results are combined with the metadata retrieved in the previous section and output to a Spark dataframe that the DatasetsSummarizer ingests.

4.4 *DatasetsSummarizer*: Exploring Dataset Search Results

In this section, we describe *DatasetsSummarizer*, a tool that enables the visual exploration and understanding of dataset search results based on similarity scores. Figure 1 illustrates the user interface. It shows *DatasetsSummarizer* being applied to compare the dataset search results for the keyword query "Taxi". The main components of *DatasetsSummarizer* are the Datasets Similarity View (B), Datasets Summary View (C), and Datasets Presentation and Exploration View (D), as shown in Figure 1. In the following, we describe each component in detail.

Datasets Similarity View. Visualization is a crucial step to get insights from data. Thorough the visualization, users can identify certain characteristics of the data, for instances, it can help to estimate which datasets are similar based on their proximity in a two-dimensional projection. *DatasetsSummarizer* provides a two-dimensional scatter plot visualization. The data is displayed as a collection of points where each point represent a dataset. Since a scatter plot needs x- and y-axis to project a point into a two-dimension visualization, we first need to generate this values. The way we do this is by computing the similarity matrix and then using the force directed graphs algorithm to compute x- and y-axis, where the proximity of the points represents the similarity between the datasets. The Datasets Similarity View supports brushing which enables users to select a subset of datasets (filtering) and analysis the different clusters that may appear in the scatter plot. If a subset of datasets is selected, all the plots in the Datasets Summary View, located on the right side on the interface, are updated accordingly.

Currently, *DatasetsSummarizer* generates by default four scatter plots based on the computed similarity between the datasets with respect to four attribute: title, description, a combination between title and description, and column names. To switch between the scatter plots, users can use the dropdown menu located on the left-top side of the visualization.

Custom Similarity Metrics. Users may want to define and use new similarity metrics to compute the similarity between datasets. To be able to make *DatasetsSummarizer* consume this new information, users just need to pass this as an additional parameter. Note that users need to add two new columns, which represent x and y values based on the new similarity matrix, as part of the input data. For example, in Figure 3, we added $modif\_title\_x$ and $modif\_title\_y$ columns as x and y values to generate a new scatter plot based on a similarity metric using a modified version of the title.

Datasets Summary View. *DatasetsSummarizer* provides a different plots to summarize the collection of datasets selected by the user. Note that these datasets represent the dataset search results for a specific query. If no selection was made, then these plots use all the datasets data. Datasets Summary View is located on the right side of the interface, as shown in Figure 1. It displays a horizontal bar chart to visualize the frequency of words within the dataset's description and title. It also provides some statistics about the whole dataset search results, for example, the number of categorical, temporal, and spatial attributes. Furhermore, the Datasets Summary View displays a parallel coordinates as another way of visualizing multiple attributes together, in this case the size, temporal, spatial

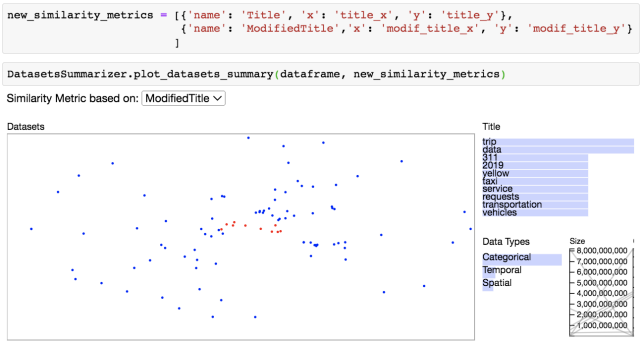

Figure 3: Add a new scatter plot based on a custom similarity metric.

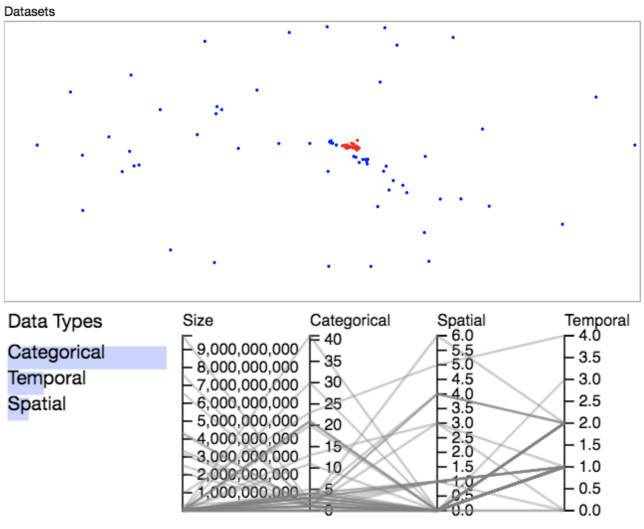

Figure 4: Parallel Coordinates to visualize multiple attributes together (the size, temporal, spatial and categorical attributes) for a subset of datasets.

and categorical attributes, as shown in Figure 1. In this visualization, points are represented as connected line segments. Each vertical line represents one data attribute. One complete set of connected line segments across all the attributes represents one dataset. Hence points that tend to cluster will appear closer together. Just by looking at it, we can clearly see whether a pattern can be observed or not. For instance, in Figure 4, we can clearly see that most of the selected datasets have low number of categorical attributes (between 0 to 5), a spare number of spatial attributes distributes between 0 to 6 attributes in total, and we can also see that most of the datasets have either 1 or 2 temporal attributes.

Datasets Presentation and Exploration View. After selecting datasets through the scatter plot, users will see the list of selected datasets displayed as snippets on the left-bottom side of the interface, as shown in Figure 1. Users can click on each dataset's snippet to show the details of that dataset on the right side. There, the metadata associated with that dataset is presented. The tabs above the

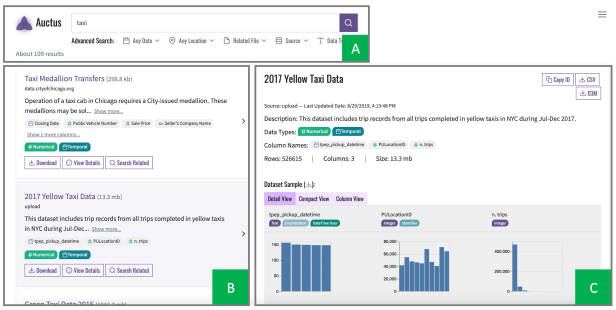

Figure 5: Auctus dataset search interface, including (A) keyword search for "taxi," which returned 109 results on May 4th, 2023. (B) List of datasets returned by search. (C) Metadata and sample for single selected dataset, "2017 Yellow Taxi Data."

table can be used to switch between the "compact view" (showing minimal information next to the column names, above a sample of rows from the data), the "detail view" (adding a plot showing the distribution of values for each column) and the "column view" (showing detailed statistical information about each column, but no sample of the values themselves).

Implementation details. *DatasetsSummarizer* is implemented as a Python 3 library. The front-end is implemented in Javascript with React [10], and D3 [5]. The back-end, responsible for data management and the Jupyter Notebook hooks is implemented in Python with Numpy [14].

The interactive visualizations were created using the following libraries:

(1) D3: https://d3js.org/
(2) datamart-profiler: https://pypi.org/project/datamart-profiler/
(3) Material-UI: https://mui.com/material-ui
(4) Python

Our system is implemented as a Python library that can be used with Jupyter Notebooks. The goal of our design is to provide a tool that can be easily installed and used by users to explore the dataset search results. We used Git (https://github.com/soniacq/DatasetsVis) to manage our system versioning and dependencies.

5 USE CASE

To demonstrate the effectiveness of our system, we will compare the process of searching for datasets with Auctus alone and the process of searching for datasets with our tool. Suppose we are interested in discovering high-traffic areas for taxi drop-offs and pickups in New York City, and how those high-traffic areas have evolved since the Vision Zero project, a citywide initiative to decrease traffic accidents, was implemented in 2014. Therefore we need to obtain taxi trip data from 2014 through the present, and this data must contain both the time and location of each pickup and drop-off.

Exploring the dataset search results for the keyword Taxi. To obtain this data from Auctus, we search for the keyword term "taxi" and receive 109 results (see Figure 5). We observe that many

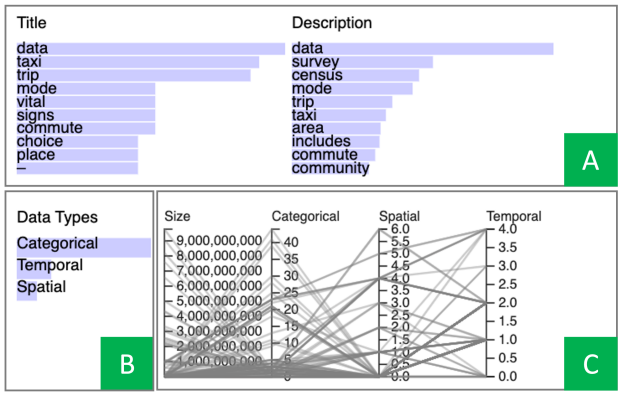

Figure 6: Portion of the DatasetsSummarizer visualization for the keyword search "taxi," including (A) the most common terms found in the titles and descriptions of datasets returned by the search. (B) The prevalence of each column type across all datasets returned by the search. (C) The size of and data types within each dataset returned by the search.

of the results are for taxi trips in New York City. However most of these only contain data for one year, for one type of taxi (i.e. green or yellow). To explore and analyze these datasets to determine, for instance, if these datasets can be merged or if there is any kind of similarity between them, we must toggle between the separate overview for each individual dataset and scroll through the sample data for a comprehensive understanding of column information. We could try to use the Auctus "search related" feature to find datasets that can merge/augment a selected dataset. However, this feature still requires us to manually inspect an overview of each "related" dataset to determine how those datasets are similar and whether that similarity is relevant to our query. Moreover, after we search for related datasets, we lose the original keyword search results. While it is possible to gather the data we seek in this manner, it quickly becomes clear that the ability to compare all of the datasets in the list of search results in a single view would facilitate the process.

We can accomplish this using our DatasetsSummarizer tool. We access the same list of 109 results for the keyword search "taxi" (again, searched on May 4th 2023) via the Auctus API. We generate the metadata for these datasets using the process outlined in Section 4.2, We then use the process outlined in Section 4.3 to compute the jaccard similarity between each pair of datasets based on their titles (breaking each into 5-shingles), descriptions (breaking each into 9-shingles), both titles and descriptions (concatenated into a single string for each dataset and broken into 9-shingles). We also compute the similarity between each pair of datasets based on their column names using MCA. We use these metrics and our DatasetsSummarizer library to generate a visualization that allows us to understand and compare the datasets in a single view (see Figure 1).

Figure 7: Views from DatasetsSummarizer visualization showing "taxi" datasets clustered by similarity based on title only, description only, and both title and description.

Before selecting anything, we see in the bar chart portion of the visualization (see Figure 6) that "trip" is the third and fifth most common term in the titles and descriptions, respectively, of our search results. This means we should have many datasets to choose from (or possibly join together) to answer our query. Moreover, while most of the columns in the datasets returned by our search contain categorical data, there are temporal and spatial columns as well (which we will need in order to determine how the high-traffic areas shift over time). Furthermore, we see from the parallel coordinates that most results have fewer than five categorical variables, and that the number of temporal and spatial variables within each dataset varies (though most have at least one temporal variable, many do not have spatial data). We can also see at a glance that most of the datasets are on the small side, with some massive outliers.

Identifying similarity between datasets. We now compare the similarity between datasets by these attributes (see Figure 7). Using the dropdown menu, we select "Title" and see that while there are some small clusters of similar datasets, this attribute does not appear to yield distinct categories. By selecting "Description" or "Title and Description" we find a much more distinct cluster of highly similar datasets surrounded by less similar outliers.

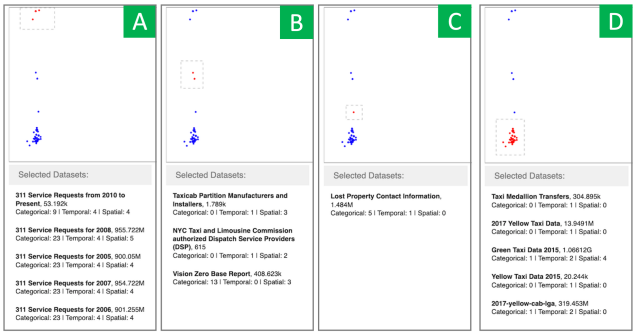

Figure 8: Views from DatasetsSummarizer visualization showing "taxi" datasets clustered by column name. We see that each cluster contains (A) 311 service request data, (B) taxi services data, (C) lost property data, (D) taxi trip data.

By brushing over this cluster, we find that it predominantly contains the taxi trip data we seek. However, this trip data is separated both by year and by taxi type (i.e. yellow or green). In order to determine whether we can join these datasets together in order to gain a comprehensive understanding of taxi trips of all types from 2014 through the present, we must determine if they have common columns. So we select "Column Name" from the dropdown menu. We note that this feature only compares dataset column names to other column names of the same type (i.e. temporal, spatial, categorical, or other).

We see that there are four main clusters when we arrange the datasets by column name similarity (see Figure 8). By brushing over the top cluster, we see that it primarily contains datasets containing 311 service request data. We can therefore disregard these because they are not related to our query.

The second cluster from the top contains a dataset about taxi manufacturers, another about dispatch providers, and the vision zero base report, all of which contain columns with information about taxi services and the areas they serve. We note the vision zero report, as it could potentially relate to our query.

The third cluster from the top contains one dataset: "Lost Property Contact Information." This dataset contains columns for values such as phone number, website, and DMV plate number. These do not seem to match any of the previous datasets, confirming its status as an outlier.

The fourth and largest cluster contains our taxi trip data; we would expect columns of the same type to be similar for trip data. For instance, many of these datasets have temporal columns containing pickup and drop-off times. Many also contain spatial columns denoting the location of the pickup and drop off. Additionally, this similarity-based overview allows us to observe that all of the datasets with titles of the format "[Year] Yellow/Green Taxi Data" have almost identical columns and can therefore be easily merged. We can also more easily identify gaps in the available data. With the Yellow and Green Taxi data, for example, we observe that while there are green taxi trip datasets for each year from 2014 through 2021, there is no yellow taxi trip dataset for 2016.

6 CONCLUSIONS AND LIMITATIONS

From our testing, the Datasets Summarizer is a useful tool that is able to quickly provide valuable information to the user that it would be onerous for them to retrieve of deduce themselves. When conducting a keyword search for "taxi", for example, the histogram and parallel coordinates make it easy to tell whether we've found any promising datasets. In our use case example, we wanted to find datasets related to taxi trips that contained spatial data. With the DatasetsSummarizer tool a task that would have required a manual one-by-one inspection of the results is instead presented in an easy to read interactive format.

The clustering visualization makes it easy to drill down into potentially interesting groupings and identify related datasets. With this tool, we were able to quickly narrow down candidate datasets to a select clumped group. It is also trivial to inspect the outliers and see which ones we want to dismiss, like the 311 datasets in our testing. Without a tool to identify outliers, we might select a dataset and invest time in comparing others to it only to find that there's very little data that we can use alongside it.

One limitation of this tool is that similarity must be pre-computed in order for the UI to be interactive. This means that though this tool would work for any datasets we selected to pre-compute those on and all of their subsets, any keywords that we did not do this pre-computation for, and any keywords whose results span the results of multiple keywords would not support this feature. In addition, if a new dataset is added it won't be accounted for in the results until they are re-computed. One potential workaround for this is a combination of some pre-computing alongside setting user expectations. If the user's search produced a subset of an existing pre-computed set, we would first return that data to the user and give them the option to re-compute it if up-to-date accuracy is required for their use case. The similarity calculations, though not fast, don't take such a prohibitively long time that the user would be unwilling to wait on them if they knew to expect an extended wait. This would be a similar user experience to the "Related File" search that Auctus provides, which searches for datasets that are joinable or unionable with the one provided by the user which is a similarly expensive calculation. Analysis could be done to determine common keyword searches made in Auctus in order to meet most use cases. In the case where the results of a user's search are not a subset of a pre-calculated set, we would notify them and give them the option to generate the results, again setting the expectation that there is an expensive calculation being made on their behalf.

The accuracy and usefulness of our similarity measurements are also limited to how accurate and useful the metadata provided by data publishers is. In this project, we saw a wide range of description detail and formatting. Some publishers for example included only a list of keywords whereas others were comprised of full paragraphs.

7 FUTURE DIRECTIONS

Our eventual goal is to integrate this tool into Auctus, so that it can work in tandem with its existing tools for dataset search and comparison. In the work described in the Methods and Data sections of this paper we have made efforts to conform to the existing structure of Auctus and use many of its tools while still providing an external proof of concept.

If our tool is integrated with Auctus, it can also benefit from any metadata additions made in Auctus. For example, allowing users to manually tag datasets with keywords could provide an interesting and useful vector for comparison. Google Datasets provides information about scholarly articles that cite specific datasets, which is a parallel effort to their dataset tagging[11]. This information could be used to provide a market-basket style analysis on a set of datasets to analyze any that have frequently been cited alongside one another.

REFERENCES

[1] [n. d.]. Estimating Similarity of Two or More Sets. https://docs.snowflake.com/en/user-guide/querying-approximate-similarity

[2] [n. d.]. Find open datasets and Machine Learning Projects. https://www.kaggle.com/datasets

[3] [n. d.]. sameAs - Schema.org Property — schema.org. https://schema.org/sameAs. [Accessed 08-May-2023].

[4] Rajeev Agrawal, Anirudh Kadadi, Xiangfeng Dai, and Frederic Andres. 2015. Challenges and opportunities with big data visualization. In *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems* (Caraguatatuba Brazil). ACM, New York, NY, USA.

[5] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D3: Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics* (2011). http://vis.stanford.edu/papers/d3

[6] Dan Brickley, Matthew Burgess, and Natasha Noy. 2019. Google Dataset Search: Building a search engine for datasets in an open Web ecosystem. In *The World Wide Web Conference* (San Francisco CA USA). ACM, New York, NY, USA.

[7] P. Brodersen. 2023. Netgraph. https://github.com/paulbrodersen/netgraph.

[8] Huan Cai, Hongyu Chen, Tie Yi, Caitlin M Daimon, John P Boyle, Chris Peers, Stuart Maudsley, and Bronwen Martin. 2013. VennPlex–a novel Venn diagram program for comparing and visualizing datasets with differentially regulated datapoints. *PLoS One* 8, 1 (Jan. 2013), e53388.

[9] Sonia Castelo, Rémi Rampin, Aécio Santos, Aline Bessa, Fernando Chirigati, and Juliana Freire. 2021. Auctus. *Proceedings VLDB Endowment* 14, 12 (July 2021), 2791–2794.

[10] Artemij Fedosejev. 2015. *React. js essentials.* Packt Publishing Ltd.

[11] Martin Fenner, Mercè Crosas, Jeffrey S. Grethe, David Kennedy, Henning Hermjakob, Phillippe Rocca-Serra, Gustavo Durand, Robin Berjon, Sebastian Karcher, Maryann Martone, and et al. 2019. A data citation roadmap for Scholarly Data Repositories. *Scientific Data* 6, 1 (2019). https://doi.org/10.1038/s41597-019-0031-8

[12] Laura M. Koesten, Emilia Kacprzak, Jenifer F. A. Tennison, and Elena Simperl. 2017. The Trials and Tribulations of Working with Structured Data: -A Study on Information Seeking Behaviour. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 1277–1289. https://doi.org/10.1145/3025453.3025838

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[14] Stéfan van der Walt, S Chris Colbert, and Gael Varoquaux. 2011. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering* 13, 2 (2011), 22–30.