Accident-Analyzer: Understanding Vehicle Accident Patterns in the United States

Kyle Otstot
Arizona State University
Tempe, US
kotstot@asu.edu

Shreyash Gade
Arizona State University
Tempe, US
sgade13@asu.edu

Jaswanth Reddy Tokala
Arizona State University
Tempe, US
jtokala@asu.edu

Anudeep Reddy Dasari
Arizona State University
Tempe, US
adasari4@asu.edu

Hruthik Reddy Sunnapu
Arizona State University
Tempe, US
sreddy29@asu.edu

Nitesh Valluru
Arizona State University
Tempe, US
nvalluru@asu.edu

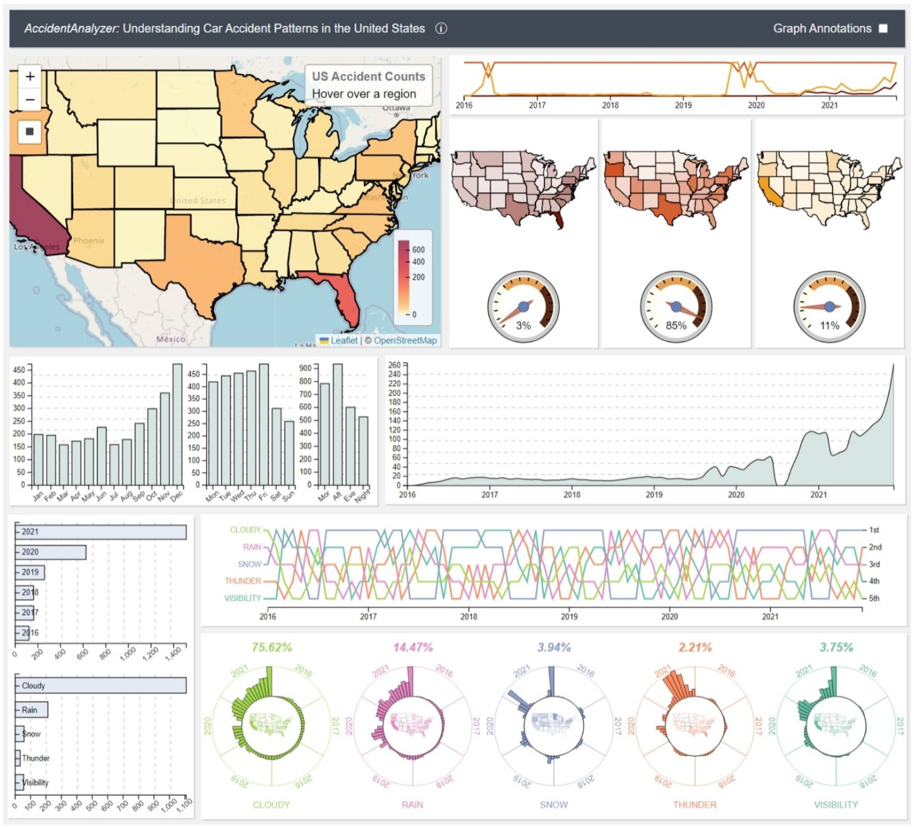


Figure 1. An overview of *Accident-Analyzer*

1 Introduction

In this project, we re-imagine CrimAnalyzer [1]- a visualization assisted analytic tool for crimes in São Paulo- in the context of traffic accidents, ultimately producing Accident-Analyzer. In doing so, we explore the spatio-temporal patterns of traffic accidents across the United States from 2016 to 2021. The Accident-Analyzer system allows for users to identify local hotspots, visualize accident trends over time, and filter the data by key weather categories in real-time. Our goals in this project were to best recreate the analytic tool proposed in the CrimAnalyzer paper, as well as extend its capabilities to the US Accidents dataset, a collection of approximately 2.8 million car accidents covering 49 states in the US. The visualization was primarily created with the *D3.js (v7)* and *Leaflet.js* JavaScript libraries, the dataset preprocessing was done using Python, and the data is stored/accessed via a MySQL database. Lastly, the hotspot computation is performed by a client-side Python library called Pyodide using Scikit Learn's non-negative matrix factorization (NMF) implementation. In this report, we look to outline the visual encodings and interactions of each section, as well as provide two case studies showcasing the usefulness of this tool.

2 Visualization Design

In this section, we detail the overall system design and interactions across eight panels (a.k.a., “views"). Each view is defined by its visual encoding, user interactivity, and filtering capabilities. We also note the added extensions.

2.1 Control Menu:

The first view spans the first row of the system, including the title of the project on the left and *graph annotation checkbox* on the right. The title also contains a link to the original CrimAnalyzer paper. When the graph annotation checkbox is selected, a tooltip appears below the cursor; as the cursor moves, the tooltip updates its content with a description of the view hovered over by the cursor. This allows the viewer to better understand the axis labels and legends, although we believe our design is intuitive enough for viewers to understand the general accident trends without the aid of the tooltip.

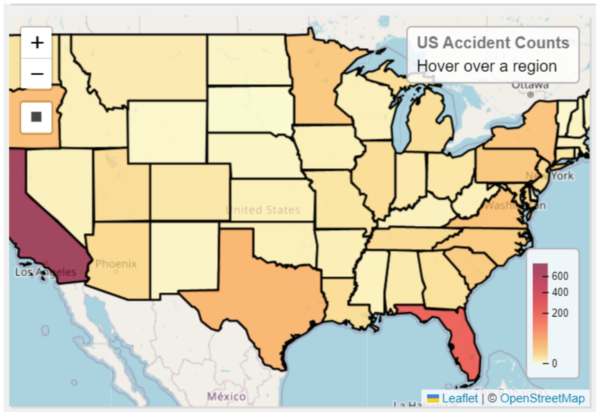


Figure 2. The initial map view

2.2 Map View:

For this view, we overlay a simple geographical map with a choropleth map that color-codes the number of accidents reported at each site in the region. In order to implement these, we used *Leaflet.js*, an open source JavaScript library designed to help developers incorporate interactive geographical maps into their web applications. As a result, we have successfully implemented the following interactions for the map view:

- **Region selection:** Users can define a region of interest by clicking or zooming onto a particular state in the map; consequently, every panel is appropriately filtered by the selected state, and this view is updated to show the county level view of the state.
- **Draw selection:** Users can leverage the panel’s *rectangle select* resource by dragging to select a rectangular region highlighting all the states or counties present in the region; consequently, the selected states or counties are highlighted in the map, and every other panel is updated appropriately based on this filter.
- **Tooltip:** When the cursor hovers over a particular state or county, we not only embolden the region but also display a tooltip in the top-right corner showing the region name and accident count.
- **Filtering:** When the region of interest is modified, ever other panel is adjusted accordingly in real time. Likewise, when the data is filtered by other panels, the coloring of the choropleth map is updated to reflect these changes. See Figures 2 and 3 for examples.

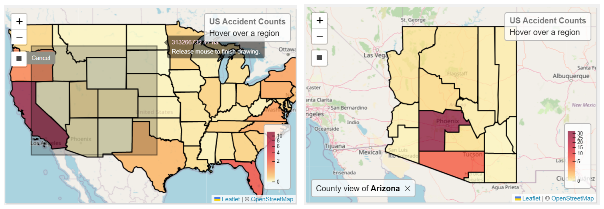


Figure 3. Region selection and county level view

2.3 Hotspot View

This view contains a collection of three hotspots, including the spatial distributions (colored map) and importance metrics (gauge) of each hotspot. The spatial distributions are color-coded to give weight (darkness) to regions with greater activity relative to the entire hotspot. Furthermore, each hotspot contains a gauge widget showing the temporal rate of occurrence of the hotspot (bottom percentage), and *measure of relevance* of the hotspot with respect to the whole set of accidents (gauge needle pointer). This measure of relevance is a bilinear interpolation of the hotspot’s temporal rate of occurrence and accident count rate relative to the other hotspots. Furthermore, our hotspot view contains a line plot showcasing the temporal activity of each hotspot over time. The original paper did not include this, so we consider this implementation a natural extension of their work. Lastly, our hotspot distributions, gauges, and line plots update in real time, which is another improvement over the original paper; for their implementation, one must manually press the play button in the control menu.

We have captured these hotspots using non-negative matrix factorization (NMF) method. We have implemented NMF in Python using the Scikit Learn library. For web implementation purposes, we used Pyodide (a Python distribution for the browser and Node.js) to compile Python code on the client side with JavaScript. Specifically, this implementation breaks the region-time matrix into matrix factors W and H, which are discussed in a later section.

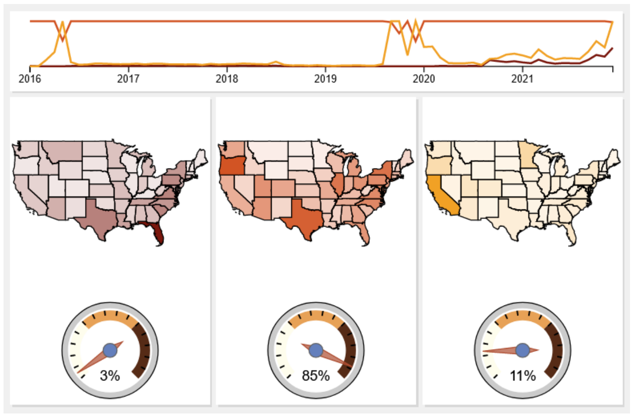


Figure 4. Hotspot view

2.4 Global Temporal View

This time-series area chart provides an overview of the number of accidents reported over the whole time period. The user can constrain the data in a particular (continuous) time interval by brushing a rectangle. When doing so, the other panels are updated to reflect the new filtered data. When a brush selection is made, the horizontal and vertical axes are properly adjusted, and a "Reset" button is provided for the user to return to the initial view, as shown in Figure .

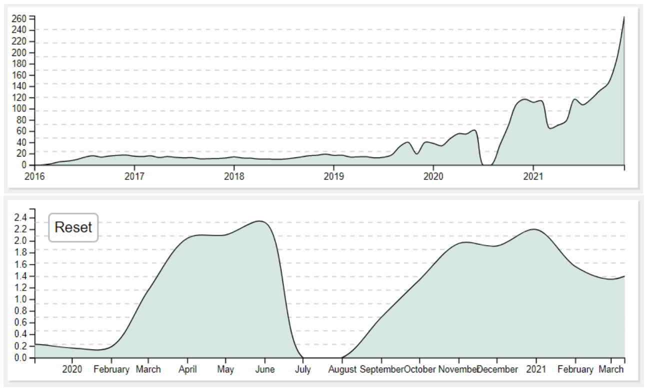


Figure 5. Global temporal view: initial view from 2016 to 2021 (top) and time interval selected by the user with the rectangular brush (bottom).

2.5 Cumulative Temporal View

This view is a collection of three bar charts displaying the number of accidents by month, day of week, and time of day (i.e., morning, afternoon, evening, and night) respectively. This allows the user to potentially identify accident patterns in cyclical, non-continuous time intervals. When other panels are used to filter the data, the bars are animated and adjusted to reflect the new conditions. Likewise, if the user wants to select any bar (or group of bars) to filter the data, the opacity of the bars are also adjusted to visualize the new constraints. This action is shown in Figure 6 below, specifically with the month *August* and day *Tuesday* being the subjects of filtering.

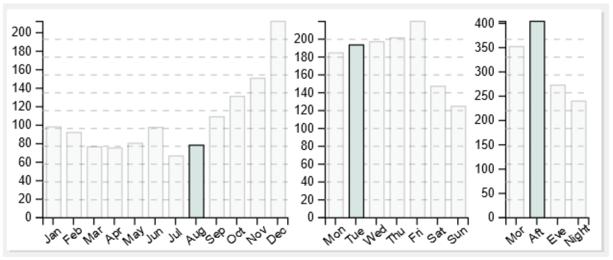


Figure 6. Cumulative temporal view

2.6 Ranking Type View

This view shows a collection of polylines that encode the relationship between 5 weather conditions– cloudy, rain, snow, thunder, visibility– and time. Each weather condition has its own polyline, with the horizontal position indicating time and vertical position indicating ranking of severity, which is defined by the length of road closure incited by the accident. When the user activates a filter from other panels, this panel is recomputed to reflect the filtered data. There are no filtering interactions for this view, but the user can hover over a weather condition to highlight the polyline and better understand the weather-severity relationship.

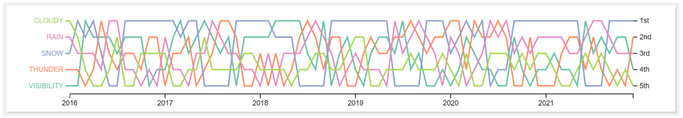


Figure 7. Ranking type view

2.7 Radial Type View

For this view, we include a collection of bar charts with a radial layout. In doing so, each radial chart belongs to a different weather condition, and the number on top shows the percentage of each weather condition in the filtered dataset. The radial bars in each chart are organized by month and year. Furthermore, the inside of each radial chart possesses a spatial distribution of the selected region (via map view) according to the accident counts grouped by weather condition. Each radial chart and spatial distribution is color-coded by weather condition, and although there are no filtering options for the user, the view will still reflect changes made by other filters in real time.

2.8 Filter Widget

This view contains two horizontal histograms over time– one for year (2016-2021), and one for the five weather conditions outlined above. The purpose of these widgets is to provide the user more ability to make quick filters based on the year and weather condition attributes. The visualization closely resembles the cumulative temporal view, and the functionality is likewise identical. Figure 8 shows two examples of selections made in these widgets.

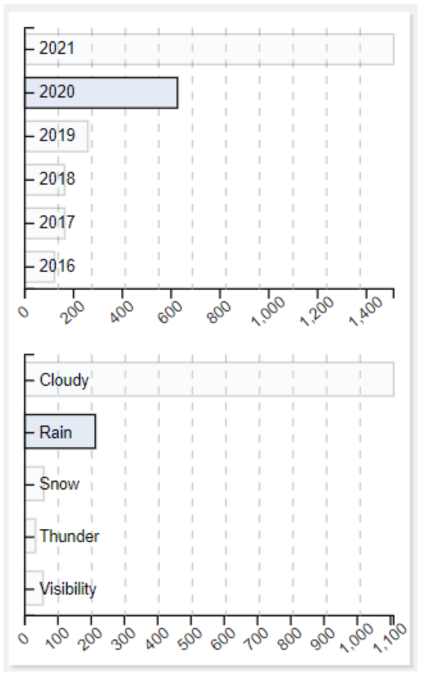


Figure 8. Filter Widget

2.9 Extensions

In this section, we briefly highlight some of the important extensions given by our visualization in comparison to the original paper's implementation.

2.9.1 Graph Annotations. As mentioned earlier, we implemented a checkbox feature which when selected will add a hovering tooltip to the interface. Whenever the user hovers over a panel, they are able to see the name of the view and a brief description of the visualization along with it.

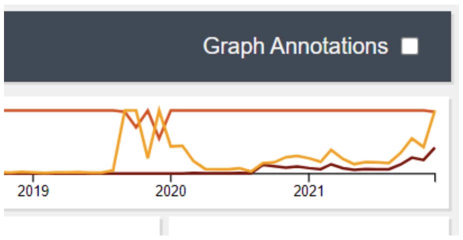
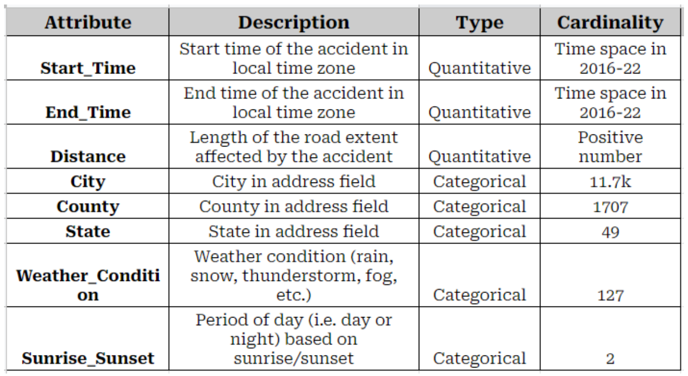


Figure 9. Graph Annotations

2.9.2 Temporal Hotspot view. Also specified earlier, this view breaks down the temporal distribution of the hotspots, where each line represents a hostpots. Moreover, when the user hovers over a particular hotspot, the corresponding line is highlighted. Everything in the hotspot view is computed in real time, which is also an extension of the original paper.

3 Dataset Description

In the paper we investigate, CrimAnalyzer queries spatial and temporal data from the São Paulo crime database; however, this database appears to not be publicly available, so instead we decided to use a US Accidents dataset from Kaggle. This dataset is of the type Table, containing 47 different attributes (columns) and approximately 2.8 million items (rows). We have not used all 47 attributes for Accident-Analyzer, since many of them are redundant or not very useful. Instead, we focused on the 7 attributes outlined below in the following table. These attributes were used to further preprocess the data for queries done by the visualization system, as described in the next subsection.



Attribute	Description	Type	Cardinality
Start_Time	Start time of the accident in local time zone	Quantitative	Time space in 2016-22
End_Time	End time of the accident in local time zone	Quantitative	Time space in 2016-22
Distance	Length of the road extent affected by the accident	Quantitative	Positive number
City	City in address field	Categorical	11.7k
County	County in address field	Categorical	1707
State	State in address field	Categorical	49
Weather_Conditi on	Weather condition (rain, snow, thunderstorm, fog, etc.)	Categorical	127
Sunrise_Sunset	Period of day (i.e. day or night) based on sunrise/sunset	Categorical	2

Figure 10. Dataset description

3.1 Dataset Preprocessing

First, the original dataset had over 100 weather categories and we coalesced and narrowed them down to the aforementioned 5 categories. Then, we calculated the duration of the accident using the *start time* and *end time* attributes of the accident. Furthermore, in order to easily query the data by non-continuous categories, we split the time stamp into various fields such as year, month, day of the week, and time of the day. Lastly, we moved the data from a CSV file to a MySQL database and hosted it locally, as specifically outlined in the GitHub Repository's README markdown. The data is queried by PHP in JSON format, and each panel receives this filtered version of the data, with the visualizations having been updated according to the new data.

4 Case Studies

In this section, we bring light to a couple notable examples of how the visualization-assisted analytic tool can serve effective use to incoming viewers.

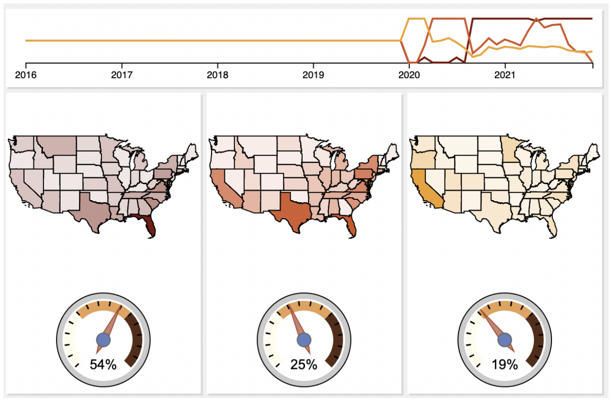


Figure 11. Visualization for the first case study

4.1 Vehicle Accidents before/after COVID-19

Here, we describe an example of noticing the difference in accident trends between some US states in terms of pre- and post-COVID lockdown. First, we use the filter widget view to select the years 2020 and 2021, which includes the time directly before and after the initial outbreak of COVID-19 in the United States. Then, we observe the hotspot view and notice that one hotspot is primarily defined by California (gold), and another hotspot is primarily defined by Florida (maroon). In the hotspot line graph, we can see that the yellow line is the highest at the beginning of 2020, but then begins to descend and ultimately falls into last place as 2020 progresses into 2021. On the other hand, the maroon line begins 2020 in last place, but increases as the year goes on and ultimately surpasses the other lines. This pattern can be indicative of how the state governments handled the outbreak of COVID-19. For example, the democratic California governor (Gavin Newsom) tended to encourage a lockdown and play it safer with the threat of COVID, which possibly caused less traffic and therefore less accidents. Conversely, the republican Florida Governor (Ron DeSantis) took an opposite stance and encouraged normal life as much as possible during the pandemic; this, in turn, likely caused relatively more traffic and therefore more accidents. Overall, this case study shows how hotspot identifications and temporal patterns can even unlock differences in ways that regions are politically governed.

4.2 Accident Severity during Winter in New York

In this case study, we show how weather conditions can yield different results in accident severity during different times of the year. First, we use the map view to select New

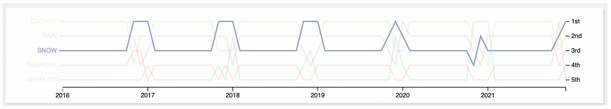


Figure 12. Visualization for the second case study

York and the cumulative temporal view to select November, December, and January (a.k.a. winter season). Then we observe the ranking type view and see that out of the 5 weather conditions, the snow condition appears to consistently cause accidents that are relatively more severe; that is, snow causes longer road segments to be blocked off after an accident occurs. Intuitively this makes a lot of sense because snowy weather makes it difficult for authorities to resolve damages and blockages caused by two or more colliding vehicles. As a result, this difficulty may lead the state authorities to close a larger portion of the affected road segment.

5 Discussion

Lastly, in this section we cover some of the lessons that we learned in this project, as well as some potential future improvements.

5.1 Lessons learned

- In this project, we learned how to set up a server using PHP and connect it to MySQL database and JavaScript to query data required for the visualization.
- Additionally, we learned how to use Leaflet.js library alongside D3.js to implement interactive maps and add filters to the selected regions.
- We also learned how to integrate Python on the client side in JavaScript with the help of Pyodide.
- Lastly, we learned how the non-negative matrix factorization can be used to generate hotspots and understood how to implement the algorithm with the help of the Sci-kit Learn library.

5.2 Improvements and Future work

- **Algorithm:** Because the non-negative matrix factorization technique is dependent on the initial conditions of the optimization procedure, our method for identifying hotspots is not stable. To avoid this effect, some implementations, such as to run the method multiple times while maintaining the result with the minimum mistake, may improve the results. Although the results become more reliable after allowing the multiple run option, a more robust method may be pursued to prevent potential effects.

- **Additional data:** Additional data that can be used to enhance the understanding of the accidents. For example, factors such as accident type, vehicle type, and the existence of specific sorts of places like bars or pubs, among other information, may have a relationship with certain types of incidents and might be utilized to acquire a better understanding of data. We may also include a panel that examines how certain incidents influenced subsequent accidents in the same location.

- **Global data:**Another enhancement would be to include worldwide data and develop the system for the entire world in order to compare accident patterns across various countries and analyze how different approaches and regulations effect accidents.

6 Additional NMF Details

We use the non-negative matrix factorization (NMF) method to help uncover spatio-temporal encodings of potentially-relevant hotspots in the data. Suppose X is a $m \times n$ matrix with each row corresponding to a region and each column corresponding to a time slice. The goal of NMF is to decompose X as a product WH where W, H are non-negative matrices with dimensions $m \times k$ and $k \times n$, respectively. In doing so, column $i \in [k]$ in W and row $i \in [k]$ in H correspond to the same hotspot i (out of k total hotspots). The rows of W indicate how much each region contributes to each hotspot, while the columns of H indicate the frequency of each hotspot over time. This allows for more intuitive groupings of regions containing similar accident patterns over time.

References

[1] Garcıa, Germain and Silveira, Jaqueline and Poco, Jorge and Paiva, Afonso and Nery, Marcelo Batista and Silva, Claudio T. and Adorno, Sérgio and Nonato, Luis Gustavo, *CrimAnalyzer: Understanding Crime Patterns in São Paulo*. IEEE Transactions on Visualization and Computer Graphics, 2021