Datenbank Spektrum (2017) 17:207–222
DOI 10.1007/s13222-017-0263-8

SCHWERPUNKTBEITRAG

The First Data Science Challenge at BTW 2017

Pascal Hirmer[1] · Tim Waizenegger[1] · Ghareeb Falazi[1] · Majd Abdo[1] · Yuliya Volga[1] · Alexander Askinadze[2] · Matthias Liebeck[2] · Stefan Conrad[2] · Tobias Hildebrandt[3] · Conrad Indiono[3] · Stefanie Rinderle-Ma[3] · Martin Grimmer[4] · Matthias Kricke[4] · Eric Peukert[4]

Received: 10 June 2017 / Accepted: 13 July 2017 / Published online: 24 July 2017

Abstract The 17th Conference on Database Systems for Business, Technology, and Web (BTW2017) of the German Informatics Society (GI) took place in March 2017 at the University of Stuttgart in Germany. A Data Science Challenge was organized for the first time at a BTW conference by the University of Stuttgart and Sponsor IBM. We challenged the participants to solve a data analysis task within one month and present their results at the BTW. In this article, we give an overview of the organizational process surrounding the Challenge, and introduce the task that the participants had to solve. In the subsequent sections, the final four competitor groups describe their approaches and results.

Keywords BTW 2017 · Challenge · Data science · Analytics · New York City · Citibike · Car accidents

✉ Pascal Hirmer
Pascal.Hirmer@ipvs.uni-stuttgart.de

[1] IPVS, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany

[2] Datenbanken und Informationssysteme, Heinrich-Heine-Universität Düsseldorf, 40225 Düsseldorf, Germany

[3] Research Group Workflow Systems and Technology, Universität Wien, 1090 Wien, Austria

[4] Big Data Competence Center (ScaDS Dresden/Leipzig), Universität Leipzig, 04109 Leipzig, Germany

1 Overview of the Data Science Challenge

1.1 Schedule and Organization

The overall organization of the challenge began one year before the BTW conference with the first of four phases:

Phase 1: Preparation task and applications. We decided to first publish a preparation task that participants had to solve in order to apply for participation. We announced that the actual challenge task will be similar, so that participants could use this preparation in order to develop their approach and choose the best suited technologies. The participants then submitted short proposals detailing how they solve the preparation task. We included this preparation and proposal phase because of the following four benefits:
(i) It allowed us to evaluate the applicants and decide which teams to accept. (ii) We expected a lower drop-out rate in the later phases if participants already invested effort into solving the preparation task. (iii) We expected higher quality contributions if the participants already had time to plan and prepare their approach. (iv) We expected to attract more participants because the preparation task reduces their uncertainty about the challenge task.
Phase 2: Reviewing applications. We received applications from seven international teams with very different approaches and invited all to participate. Three of those teams dropped out later despite the preparation phase, leaving us with four final participants.
Phase 3: Challenge task. The actual challenge task was published one month before the BTW conference, starting the main phase of the challenge.
Phase 4: Presenting the results. One month later, each team had 30 minutes to present their results in front of the conference audience and a jury. The jury comprised: Dr.

Martin Mähler (IBM, Manager University Relations), Sven Hafeneger (IBM Analytics), Andreas Tönne (CTO, Dibuco GmbH), Dr. Matthias Wieland (Uni Stuttgart, IoT/Smart Services), and Pascal Hirmer (Uni Stuttgart, Data Mashup Researcher).

1.2 Challenge Task

Traffic accidents are one of the main causes of death throughout the whole world. Reducing the number of accidents is a difficult challenge due to the many influencing factors that lead to them. To tackle this issue, the overall task of the Data Science Challenge was investigating how traffic accidents can be reduced or even avoided in the future by creating recommendations for governments or cities.

These recommendations should be generated based on the following datasets: (i) the *New York City car accidents data* is an open dataset of the city of New York that contains near-realtime information about occurring car accidents. It contains, amongst other information, the date and time of the accident, the location, the involved vehicles or persons, the cause (if known), and the number of injured or killed persons. This dataset serves as the foundation for all analyses, all other datasets are optional. (ii) The *Vision Zero DB* is a collection of infrastructure data, e. g., about street design or speed limits. (iii) The *3-1-1 callers* dataset contains complaints of citizens regarding potholes, non-working traffic lights, or noise. (iv) Corresponding weather data of New York City for the time range of the accident data is provided by the IBM weather company. This weather data contains, amongst others, the temperature, humidity, precipitation, snow, and air pressure. (v) Another dataset contains all Taxi and Uber rides of New York City in the time range of the accident dataset. All these datasets are available as CSV files. Furthermore, information about occurring events that could influence the amount of traffic can be extracted from social media such as Twitter. Map data from different platforms such as Google Maps can be used, e. g., for visualization of the results.

Based on this data, different questions were provided that serve as inspiration for the challenge participants. They could decide whether they want to do a comprehensive but shallow analysis or a narrow and deep analysis that focuses on one question.

- Visualize the data and reveal underlying patterns:
  - Are there accident-free/exceptionally safe roads?
  - Where are the most dangerous spots?
  - Can you identify classes/types of accidents? Which ones are the most frequent?
- What factors have the biggest influence on accident probability?
- Can you observe an unusual/significant increase or decrease in accidents anywhere? What actions or events caused it?
- Can you find a connection between accidents and large public events?
- Propose actions that will lower the number of accidents based on your results from above:
  - What could the city change or improve?
  - What should drivers do differently?

1.3 List of Participants

The participants of the Data Science Challenge were:

Team 1 (Universität Stuttgart): Ghareeb Falazi, Majd Abdo, Yuliya Volga
Team 2 (Heinrich-Heine Universität Düsseldorf): Alexander Askinadze, Matthias Liebeck, Stefan Conrad
Team 3 (Universität Wien): Tobias Hildebrandt, Conrad Indiono, Stefanie Rinderle-Ma
Team 4 (ScaDS Dresden/Leipzig): Martin Grimmer, Matthias Kricke, Eric Peukert

1.4 Challenge Outcome and Conclusion

The challenge's jury decided on the winner based on the following criteria: (i) impact and novelty of the results, i. e., potential for avoiding accidents, (ii) comprehensiveness and scope of the results, (iii) visual/aural representation of the results, and (iv) live-presentation at BTW on March 7.

The winning team of the challenge was Team 1 from the University of Stuttgart. They could convince the jury by presenting a powerful platform that can conduct comprehensive analytics based on the above-mentioned datasets. Especially the maturity and the provided visualizations of this platform impressed the jury. The second place went to Team 2 from the University of Düsseldorf. They developed a new kind of navigator application, similar to Google Navigation that searches for the safest instead of the shortest or fastest routes. Finally, the jury decided to have two split third places because the results of both remaining teams were promising. Team 3 of the University of Vienna presented a nice platform to navigate through the data as well as an approach for data analysis using sonification. Team 4 of ScaDS Dresden/Leipzig combined the citizen complaints data with the accident data to predict occurring accidents in the future.

The following 4 sections contain the approaches and solutions, written by the participating teams. Each team contributed 4 pages to this composed article.

2 First Place: Cloud-Based Visual Analysis of Motor-Vehicle Collisions in New York City in the Period 2012-2017

Ghareeb Falazi, Majd Abdo, Yuliya Volga

2.1 Introduction

The analysis of traffic collisions is getting more and more important in the past years. In fact, road injuries have become recently one of the top 10 reasons for death worldwide [12]. However, the analysis of traffic collisions is not a naive task as it involves many dimensions as well as multiple data sources related to the analysis. Due to this complexity, performing only automated analysis to traffic collisions data is not adequate which makes the involvement of human judgment also required. For this reason, we have decided to use Visual Analysis [5] which is, in principle, a combination of automated analysis techniques with interactive visualizations that allows to achieve a better result when the problem setting requires exploration and gradual understanding of the data and the underlying patterns. It is in fact relevant to many domains. Public safety and geospatial applications are considered two of them [10]. To this end, we have created a cloud-based application that hosts a website with multiple interactive visualizations to apply many potential analytics scenarios to the heterogeneous data sources at hand.

In this article, we introduce the method we used to integrate the data sources and the architecture of the solution. Furthermore, we introduce the set of interactive visualizations of our approach, as well as one demonstrative use case and the interesting findings we have made in addition to the possible future work.

2.2 Integration of Data Sources

Visual Analysis begins with data, which is usually heterogeneous and large. For our analysis, we use three data sources: the NYPD Motor Vehicle Collisions dataset, the Yellow Taxi Trip Data in Manhattan, and the historic weather records of the considered time frame. To prepare the data for further analysis, we followed a pipeline similar to the famous ETL process of Data Warehousing [2]. The three sources have clear schemata and after the acquisition as CSV files, we transformed them by joining the weather data with the accidents data on the date attribute, and aggregating the taxi trip records based on the hour of each trip in order to handle the huge size of the data, other cleaning and alignment tasks were also performed. To host the integrated data sources, we use PostgreSQL DBMS which allowed us to use the PostGIS add-on to build a geo-based index that facilitates quick execution of geo-based queries executed against the location of the accidents. To further enhance the performance, we built materialized views on top of the integrated tables for frequent queries with moderate-sized results. The integration pipeline was executed using a Python 2.7-based Jupyter Notebook, and the final database was made available to the cloud application as a Compose service.

2.3 System Architecture

The solution is cloud-based and it is hosted on IBM Bluemix cloud as a Cloud Foundry Python application. The application itself is a 3-tire Flask web server that communicates with the database, performs server-side analysis, builds the interactive UI and supplies it with new data when needed using AJAX. For building the visualizations and the user controls, we primarily used Bokeh, a rich open-source Python library that provides an extensive list of customizable visualizations as well as many interaction possibilities using filtering and selection that initiate asynchronous callbacks to the server to retrieve new and refined data to visualize. Furthermore, we use Google Maps Javascript API and Folium for map-based visualizations. These visualizations and their capabilities will be further explained in the upcoming sections.

2.4 User Interface

The application interface consists of six main interactive visualizations (described below). These visualizations are shown as separate tabs in the user interface.

Basic Statistics — is a visualization which gives the user general insights on the data. There you can find the distribution of accidents over city boroughs and seasons, the number of accidents per vehicle type, and the top reasons of the accidents.

Accident Locations — is a visualization which creates a heat map [7] on which the locations of different types of accidents can be shown. Various filters can be applied on the data which facilitates gradual exploration of hidden patterns and interesting details. These filters are: specifying the period of time considered, and filters to specify the day of week, the reason of accidents, the borough, the vehicle type, the minimum number of accidents and the minimum number of causalities to consider. The weighting of the heat map points can be based on either the overall number of accidents occurred at each point, or the number of casualties associated with the accidents. With this visualization you can discover, e. g., where accidents caused by the reason ‘‘Glare” happened and if death cases were reported, or the most dangerous spots in the region.

Accidents over Time — a visualization, which shows three line charts:

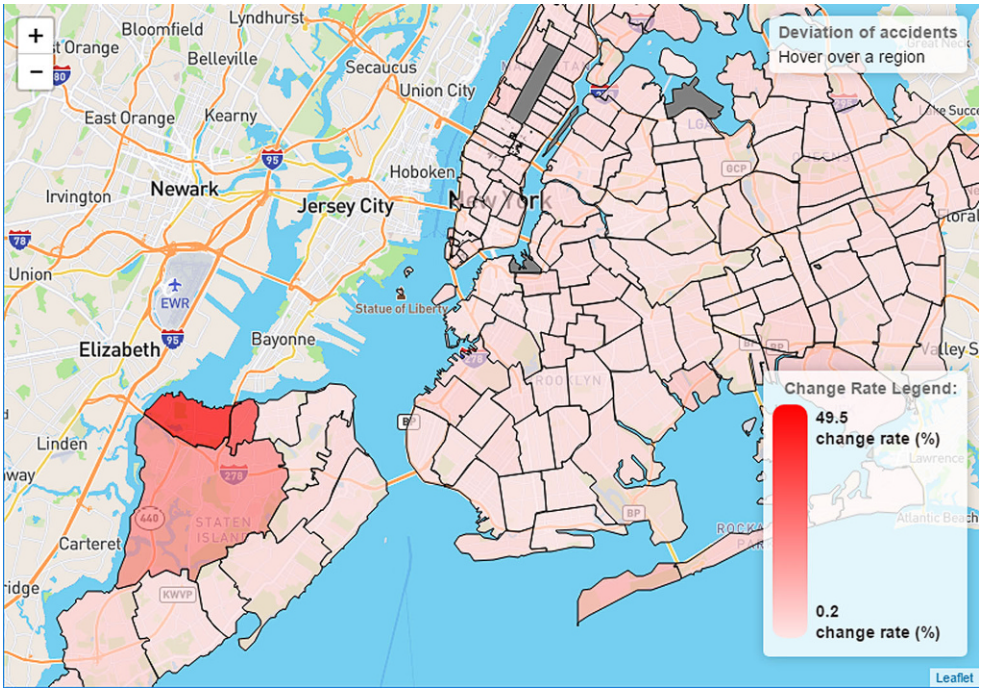

Fig. 1 Change rate values of NYC ZIP-code areas. (darker red means higher change rate)

(i) a time-line with the number of accidents over time,
(ii) the components resulting from time-series decomposition [1] of (i), and
(iii) the number of accidents per hour of the day accumulated for all days within the chosen time period.

In this interactive visualization, filters can also be applied to focus on a specific period, a reason of the accident, or the vehicle type. What we find to be particularly helpful are the ''seasonal" and ''trend" components of the time-series decomposition (ii) as they allow the user to detect interesting patterns and tendencies in the data especially if they are combined with variant filters.

Change Rate — is an interactive visualization, which shows a choropleth map [9] divided by the ZIP-code areas of New York City. The color code of each ZIP-code is assigned a value that reflects the change rate of the number of accidents over time which is calculated based on either consecutive years or months. It is precisely described by the following formula:

$$\frac{\sum_i^n |x_i - x_{i-1}|}{\sum_i^n x_i} \cdot 50 \qquad (1)$$

n: number of years/months under consideration
x_i: number of accidents at the ith year/month
x_{i-1}: number of accidents at the $(i-1)$th year/month
Multiplication by 50: to make the result range between 0% and 100%.

Using this tool, we discover regions in the city which have sudden increases or decreases in number of accidents — this helps in pointing the user to where it is interesting to further discover other aspects of the data.

Relationship to Weather — a visualization with line charts describing the effect of weather conditions on the number of accidents. For instance, bad visibility conditions are associated with a high number of accidents. Here, for each weather value, the number of accidents is accumulated and divided by the number of days with the corresponding value for normalization.

Relationship to Traffic — the assumption behind this visualization is that the number of taxi trips could be a good indicator of traffic in NYC[1]. Thus, the dependency between the traffic level and the number of accidents might be discovered. In fact, we noticed a strong positive correlation between the traffic level and the number of accidents.

2.5 Demonstrative Use Cases

In addition to the tools introduced above, the user might be also interested in disclosing further details not presented in a direct way. For this purpose, multiple visualization tools can be combined to investigate some distinguishable cases, e. g., a distinctive peak or drop of accidents in a specific zone/period.

One scenario the user might follow is to overview the top-3 vehicle types causing accidents from *Basic Statistics* (Sect. 2.4), then by using the *Change Rate* visualization, the user can easily see the overall (yearly/ monthly) change

[1] Especially because, in Manhattan alone, about 2.8 million taxi trips take place each week [8]

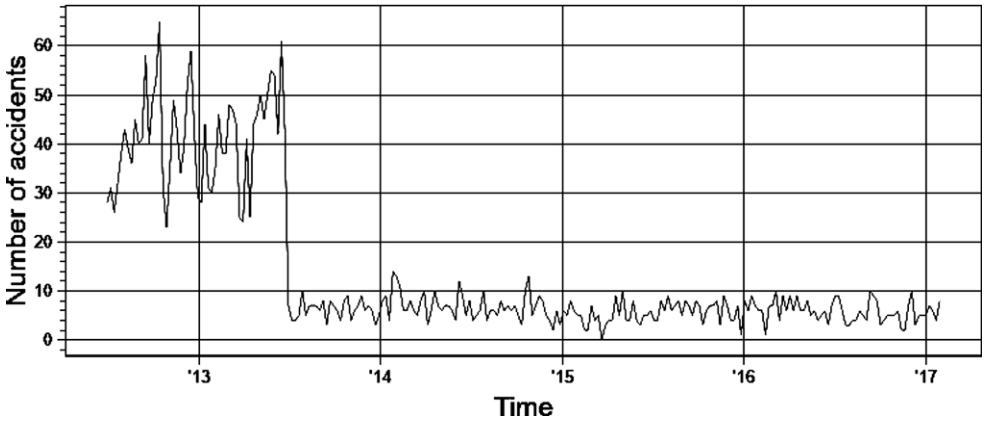

Fig. 2 Historical timeline of accidents for one ZIP-code area in Staten Island that shows a high change rate

rate for all ZIP zones of NYC and specifically for those vehicles types.

A portion of Staten Island borough, as an example, is colored in dark red, since it has witnessed a distinct decreasing rate in number of accidents over the recent years (Fig. 1). However, this information could be insufficient alone and the user would like to investigate for additional details. Then, using the *Accidents over Time* visualization helps to see explicitly the historical change (Fig. 2). More advanced analysis would be decomposing the time series into trend and seasonal components which allows to clearly see the data behavior. By executing the above scenario, all the details of when and where the changes in number of accidents have happened, can be easily collected.

This approach of visual analysis, which we used to discover insights, was introduced originally by Keim [5] as a mantra called "Analyze first, Show the Important, Zoom, filter and analyze further, Details on demand" that is implemented by combining analytical approaches (e. g., time-series decomposition) together with advanced visualization techniques (e. g., choropleth).

2.6 Important Findings

By using the application's tools, we were able to discover many interesting insights and suggest helpful recommendations to reduce accidents in the future. In the following parts, we introduce some of those findings which cover the period 2012 to February 2017.

Concerning accident numbers, "Flatbush external to Manhattan bridge" has been recorded as the spot with the highest number of accidents (precisely 678). In addition, Linden Boulevard in Brooklyn has witnessed the highest number of casualties (injured or killed) in the whole city (183 in total). It is also worth to mention that accidents caused by over-speeding have increased tremendously since September 2016. Therefore, special attention from drivers and strict policies from the NYC municipality side should be taken to reduce this risk.

As one might not expect, the most extreme weather conditions are not associated with a high number of accidents; our assumption is that people are over-cautious when certain weather conditions take place. But surprisingly, less extreme values have the highest rate of accidents, thus, we recommend that the media plays a role in encouraging people to drive more carefully whenever certain ranges of extreme weather values are likely to occur. Moreover, pavement slippery appears to be one of the top reasons of accidents among the infrastructural ones, especially during the Winter. Therefore, work and actions need to be done on roads to prevent such incidents. Finally, we noticed a significant degradation of data quality in the process of recording the reason of accidents starting from April 26th, 2016, and even though this was partially recovered starting from September 8th, 2016, NY Police Department should investigate closely to tackle such an issue in the future.

2.7 Future Work

There are several further aspects that are interesting to try with this set of data sources which can give new insights or enhance the current ones. For example, showing the choropleth map of the change rate based on the ZIP-code areas is easy to implement, but not necessarily perfect, a more natural clustering of accidents could be based on a spatio-temporal clustering scheme [6]. Furthermore, applying data mining and machine learning techniques on the given dataset is another option for disclosing interesting patterns and behaviors. For instance, predicting the number of accidents for a certain area and time using regression and classification techniques can help to determine the number of ambulance crews that should be prepared for emergencies. Last but not least, expanding the number of data sources to combine with the primary accidents data source can unveil new results. For example, using a data source about speed limits[2] can help in checking the correlation between num-

[2] http://www.nyc.gov/html/dot/html/about/vz_datafeeds.shtml

ber of accidents and speed limits in particular regions, and tweet messages about city events may reveal new correlations as well.

3 Second Place: How to Travel Safely in NYC?

Alexander Askinadze, Matthias Liebeck, Stefan Conrad

3.1 Introduction

In this chapter, we describe our contribution to the SDSC17 challenge in which we achieved 2nd place. Given the *NYPD Motor Vehicle Collisions* dataset about car accidents in New York City, we were asked to explore and analyze the dataset. Some tasks and questions were suggested, for instance: (i) where are dangerous spots?, (ii) where are accident-free spots? (iii) visualization of the data, (iv) creation of an animation of the development over time, and (v) the analysis of descriptive statistics and correlations. The latter include (a) what types of accidents occur?, (b) are there connections between accidents and large public events?, and (c) what factors influence accidents? With our contribution, we mainly focused on answering the question ''How large is the potential for avoiding accidents?"

The participation in the challenge required the usage of cloud technologies. We decided to use Microsoft's cloud technology called *Azure*[3]. In addition, Microsoft currently offers a free cloud-based machine learning platform called *Microsoft Azure Machine Learning Studio*[4] (*Azure ML*). Azure ML provides drag and drop functionality to process datasets and to create machine learning pipelines with steps such as data cleaning, data splitting, and training and evaluating classifiers, e. g., support vector machines (SVM) [3] and random decision forests [4]. In our approach, we used a combination of Azure and Azure ML to analyze and visualize the data and to create a Web-based prototype that offers less risky travel routes through New York City.

3.2 Data Analysis and Statistics

For a first overview of the data, the dataset was uploaded to Azure ML. The dataset consisted of about 1 million rows. Since some rows had no GPS coordinates of the accidents, we performed a data cleansing with an Azure ML component which resulted in 770 thousand remaining rows. Afterwards, it was immediately possible to create first exploratory statistics of the dataset's attributes, such as *number of persons killed* and *number of persons injured*. Table 1 shows statistics which are calculated over the cleaned data for the attribute *number of persons killed*.

Table 1 Azure ML statistics: *number of persons killed*

Mean	Median	Min	Max	STD	Unique Values
0.0012	0	0	5	0.0361	6

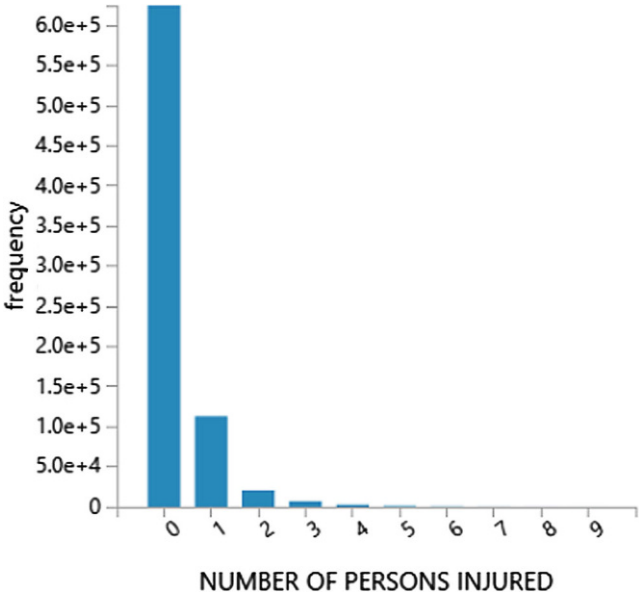

Fig. 3 Number of persons injured

The evaluations are given not only in numerical form, but Azure ML also provides bar and boxplot diagrams. Figure 3 shows that, fortunately, no persons were injured in 84.4% of the reported accidents and that approximately 13% of the accidents resulted in the injury of one person.

Fig. 4 visualizes specific days and their accident frequencies on which the most accidents happened. We cross-referenced these days with news articles to find out, if specific major events might be responsible for the higher number of accidents. The increased accident rate on January 21, 2014, is most likely related to a snow storm that hit New York city on that day[5]. The day with the second most accidents reported, February 2, 2014, was the day after the 2014 Super Bowl.

By analyzing the individual accident locations, some places turned out to be more accident-prone and that most of the accidents happen in Brooklyn, Manhattan, and Queens.

Upon analyzing the contributing factor for the first involved vehicle, we see that over 50% of the accidents do not have a contributing factor in their accident report. The most common (specified) cause is distraction, followed by being fatigued.

[3] https://azure.microsoft.com

[4] https://studio.azureml.net

[5] https://www.nytimes.com/2014/01/22/nyregion/east-coast-snowstorm-takes-aim-at-new-york-region.html

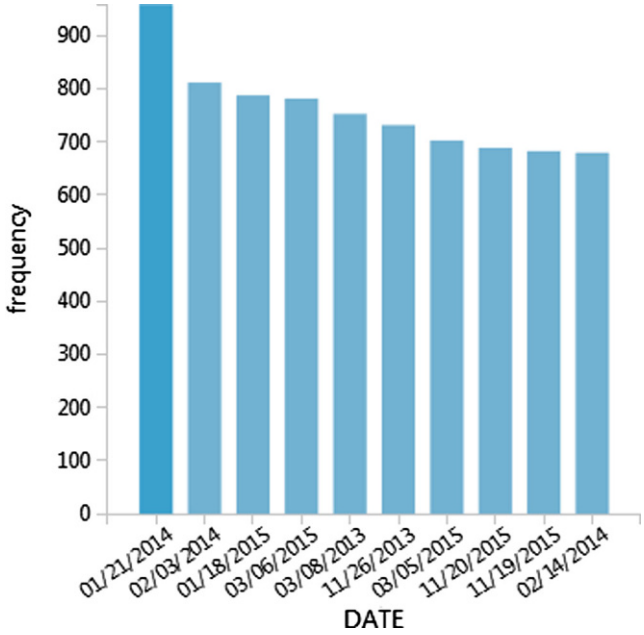

Fig. 4 Accident frequency per day ordered by quantity

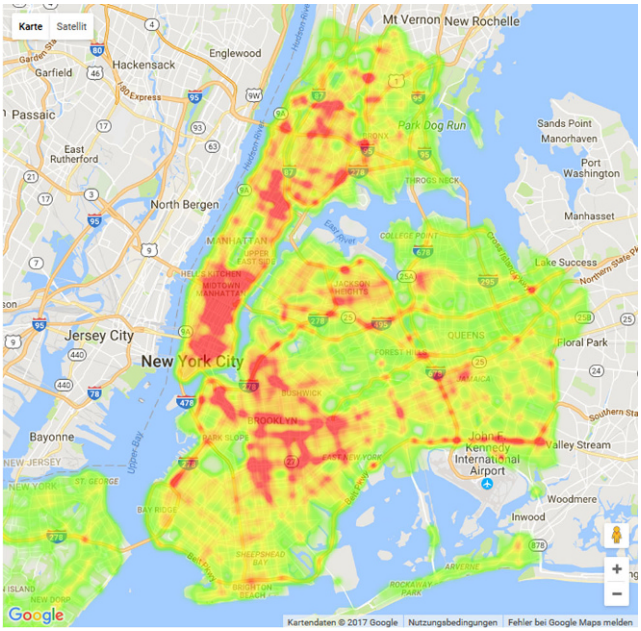

Fig. 5 Frequency heatmap, map data (© 2017 Google)

3.3 Visualization

In order to visualize the coordinates of the accidents with their corresponding frequencies, we used the Google Maps Heatmap Layer[6]. A heatmap is able to render *WeightedLocation* objects that contain the information about a specific point on the map and a weight. A higher weight will cause the WeightedLocation to be rendered with a greater intensity than a smaller weight. In this way, the number of accidents in different locations can be better represented, as shown in Fig. 5 which visualizes that most accidents occurred in Manhattan.

Since the number of accidents depends on many factors, such as the vehicle type or the date and time, we implemented a filter functionality for the underlying data. This allows us to visualize these different factors on the heatmap, which can be seen on our project site[7].

In order to better understand the time dependency of the accidents, we implemented an animation for a time frame between a selected start and end date.

3.4 Routing

After visualizing the accidents on a heatmap, we recognized that certain areas are more dangerous than others in terms of accident frequency. A viable approach to reduce the number of accidents is to simply avoid these areas (which are always intersections, since the GPS coordinates are rounded to the nearest intersection) and to circumvent them with an alternative route. This solution benefits from New York City's grid street plan.

Therefore, we proposed a Web-based navigation software for the following scenario: Given a start location and an end location, we wanted to plot a route that is as safe as possible while respecting the means of transportation $t \in \{\text{pedestrian}, \text{bicycle}, \text{car}\}$.

We developed a prototype by using three external APIs: (i) Geoencoding from Google Maps API[8], (ii) Routing from HERE[9], and (iii) visualization with Google Maps. The workflow of the prototype for determining the fastest route is as follows:

1. Enter a start and an end location
2. Choose the means of transportation
3. Geoencode the entered adresses with Google API
4. Get the route via the HERE API as a JSON response of GPS coordinates
5. Plot the returned GPS coordinates on Google Maps

As an example, let us assume that we want to travel from the Flatiron Building to the Times Square via bicycle. The fastest route determined by the HERE Routing is illustrated in Fig. 6a.

Since the fastest route might be alongside dangerous intersections, we identified dangerous spots based on the dataset within a minimal bounding rectangle between the

[6] https://developers.google.com/maps/documentation/javascript/heatmaplayer?hl=en

[7] http://btw2017-dsc-hhu.azurewebsites.net/

[8] https://developers.google.com/maps

[9] https://developer.here.com

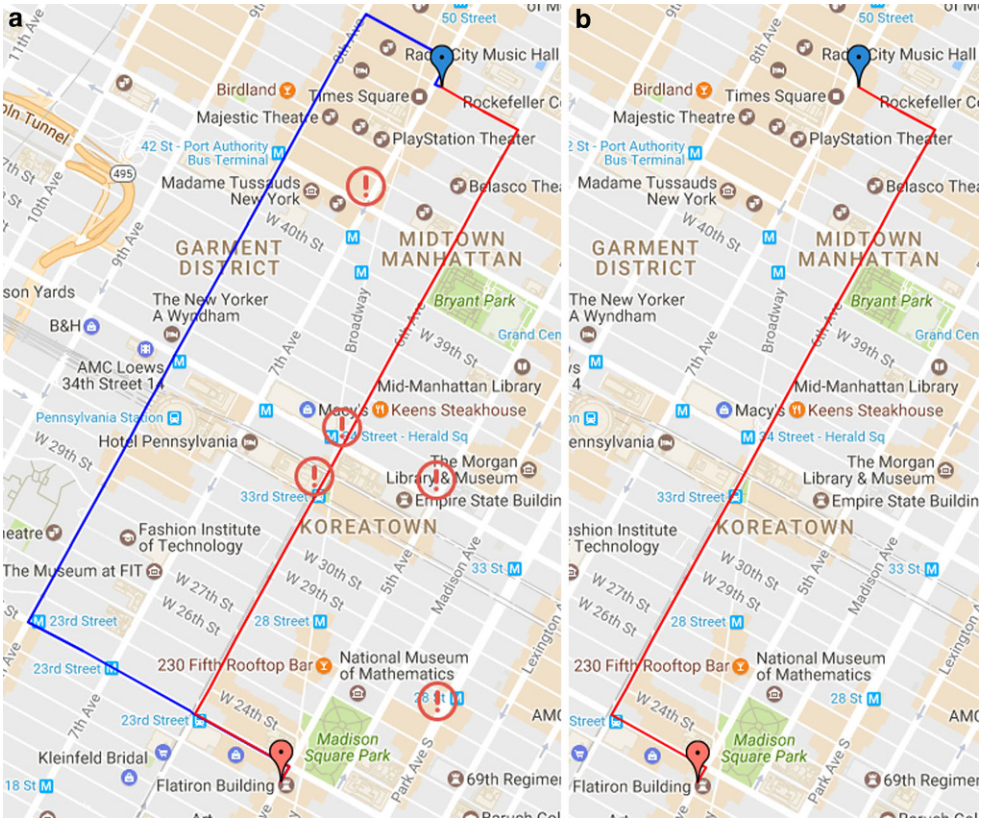

Fig. 6 Calculating routes from the Flatiron Building to Times Square, map data (© 2017 Google). **(a)** Fastest route, **(b)** Safer route (dangerous spots are illustrated by exclamation points)

start and end locations while respecting a personal risk factor (a lower value indicates a higher will to take risks). These spots are displayed on the map as exclamation points. In order to determine a safer route, we passed the dangerous spots to the routing API and specifically excluded them from the route. Figure 6b shows that the safer route includes a detour and that it circumvents two dangerous intersections while respecting the direction of travel.

3.5 Conclusion and Future Work

Our challenge contribution consists of presenting descriptive statistics in Azure ML and creating a Web-based prototype which is capable of visualizing the accidents with a heatmap and animating the development of the accidents over time. Additionally, we proposed a new routing approach that incorporates the accidents from the dataset with real-time traffic information in order to possibly make transportation in New York City a little bit safer. An advantage of our proposed navigation method is that it is low-cost. The costs depend on the hosting and the API prices. Since the solution is cloud-hosted, it is scalable even for larger cities like New York City. The only requirement for using our approach is that the city has an available motor vehicle collision dataset containing the necessary information, e. g., precise geocoordinates and time stamps, and a sufficient budget for IT costs.

In our future work, we want to include temporal aspects and weather conditions to improve the routing calculation. Additionally, we want to automatically decide whether a given route is safe enough to travel. Therefore, an accident risk is calculated for each intersection along the given route by using a regression SVM that calculates an accident risk based on geocoordinate, time of day, and vehicle type. Afterwards, the maximum of the accident risks will be compared with the personal risk factor to make a final decision.

4 Third Place: Audiovisual Analytics

Tobias Hildebrandt, Conrad Indiono, Stefanie Rinderle-Ma

4.1 Introduction

Our approach, rather than only using methods from statistics and machine learning to (semi-)automatically detect patterns, trends and correlations in the data, strongly relies on humans' capability of pattern recognition. We will make use of different kinds of percepualization to obtain a first overview of the data and its structure, see (and hear) patterns and trends and generate hypothesis that can later be verified or rejected using data mining and other means of statistical analysis. Besides interactive visualization, we

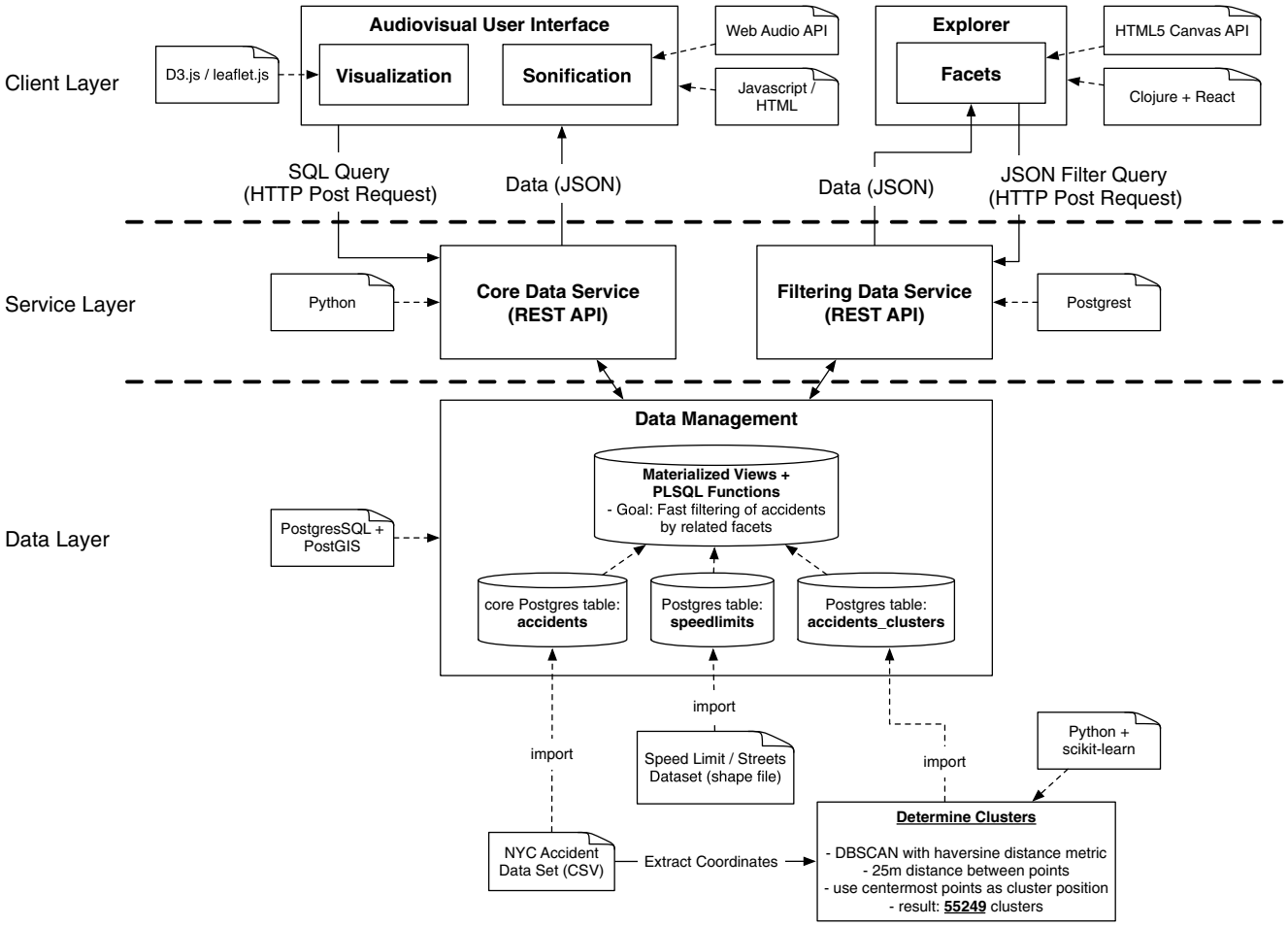

Fig. 7 Concept

will also apply methods from the area of sonification, the presentation of data using (non-speech) sound. Sound, unlike visuals, can only exist in time and we as humans are very good in detecting slight variations in sound over time, e. g., in rhythm. This advantage will be combined with the power of visualization to convey spatial data.

4.2 Data Management and Data Preparation

The architecture for realizing our audiovisual data analytics is based on a standard three-tier architecture. In this section, we introduce the service as well as the data layer (cf. Fig. 7), starting with the latter. For providing the data for the audiovisual analytic components, we have introduced a three-phase data preparation step, basing on two data sources: one being the core NYC Accident dataset (as CSV) and the other being the speed limit and streets shape file dataset. In the first import phase, we have written respective scripts to transform and import the resulting data into two PostgreSQL tables: *accidents* and *speedlimits*. The next phase, clustering, focuses on the generation of accident clusters that have a radius of 25 meters. The method for generating these clusters is DBSCAN[10] using the Haversine distance metric. To determine the cluster position, we have taken the centermost point from those accidents within the cluster. This phase generates 55249 clusters and ends with importing these into a separate cluster table, *accidents_clusters*. These clusters are referenced from the main accidents table, so that each accident has a cluster associated with it, allowing the query of spatially related accidents. The final phase focuses on generating materialized views for the purpose of fast filtering of data.

There are four goals and levels the materialized views are designed to accomplish. The first one is concerned with the fast lookup of any given accident to a facet. One facet might deal with the specific hour or week day an accident occurred, while another is responsible for knowing in which street intersection an accident occurred. Each facet is stored in its own materialized view indexed by their respective primary keys. Normalization plays an important role at this level as we want to uniquely store facets which

[10] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.dbscan.html

Table 2 Focus of modalities

Mode	Visualization	Sonification
Static	Details on individual accidents	Acoustic summary of current selection
Animated	Details on individual accidents	Development of accidents over time

are represented differently in the raw dataset. One example is the case for facets dealing with location data, concretely the normalization of street names. For example, the keyword EXPRESSWAY occurs within the same dataset as three different variations: EXP, EXPY, EXPWY. We found five normalized keywords within street names that occur with differing variations. Towards achieving the first goal of fast lookup, another level of materialized views are created for memoizing each unique *accident ids* to their respective facets. The second level goal deals with the fast lookup of shared statistical metrics. The metrics we track for accidents are: the number of accidents and the number of persons who are either injured or fatally killed. The latter are further split into categories: pedestrians, cyclists and motorists. This level of materialized views store the respective metrics for each *accident id* allowing for later aggregation. The next level goal concerns the efficient filtering of various facets, resulting into a set of *accident ids* to be fed to the materialized views from level 1 and 2. This level is accomplished using a custom PLPGSQL function. The fourth and final goal is to build a cache for the most common filters, pre-computing the most expensive filter results. This allows the user interface to act as fast as possible, avoiding waiting time for the user once she starts filtering the whole dataset. The most common variation of filters three levels deep are pre-cached into such materialized views.

Once the materialized views are pre-generated, the data preparation stage is marked as complete. From this point on, the service layer is responsible for exposing the prepared data for consumption via a REST API. This is split into two services as the two different clients focus on diverging datasets. Whereas the final core PostgreSQL tables are emitted in the Python-based *Core Data Service*, the Postgrest-based *Filtering Data Service* mainly accesses the materialized views and specialized PLPSQL functions for quickly filtering the accident data.

4.3 Explorer

The goal of the Explorer, as the name suggests, is to give the user a visual explorative method for drilling down the data. This is accomplished by allowing the user to filter the accident data by casualty type (e. g., persons injured, cyclists killed) and by various facets. These facets can be based on time (year, month, day of week, hour of day), location (boroughs, street intersections, addresses), vehicle type, accident factors (e. g., inattention) and spatial clusters. All of these facets have their unique representations and allow further combining by these facet attributes to facilitate quick visual data exploration, without having to manually design and submit complex SQL queries. One example exploration might filter the data where accidents involve pedestrians being injured (casualty type), the accidents happening between the months of July to September (time), where taxis are involved (vehicle type), and the accidents are situated in Manhattan (location). Each facet is realized as a component using the Clojure programming language and the React UI library, allowing reuse of common display logic and hiding of state. The backbone for facilitating the (1) mixing and matching of filters (accident factors and facets) as well as the (2) fast querying result are the previously prepared materialized views. These and specific helper PLPGSQL functions are exposed as REST API endpoints with the help of Postgrest[11]. Each component takes such a REST endpoint as the source of its data and then renders the data as appropriate, allowing fast data exploration. The source code for the explorer front-end and the data management back-end can be studied from this link[12].

4.4 Visualization and Sonification

The user interface is built in such a way that its main purpose is to provide an overview over the data and the distribution of accidents (both spatial and temporal), but present detailed information on individual accidents on demand. As the dataset is inherently time-based, the UI also offers an animated mode that represents developments over time. In both modes, the visualization presents concrete details, while the background sonification presents general trends (Table 2).

In order to enable the users to filter and navigate the data in real-time, several checkboxes and sliders enable to, e. g., select the timeframe or filter by certain accident causes (cf. Fig. 8). In the following, the data attributes represented are listed, together with the visual (and acoustic) attributes they are mapped to. It is planned to let the user customize the mapping to visuals and sound, but this feature is not yet implemented.

- Accident location: circle position

[11] http://postgrest.com

[12] https://github.com/indygemma/nyc-accident-explorer

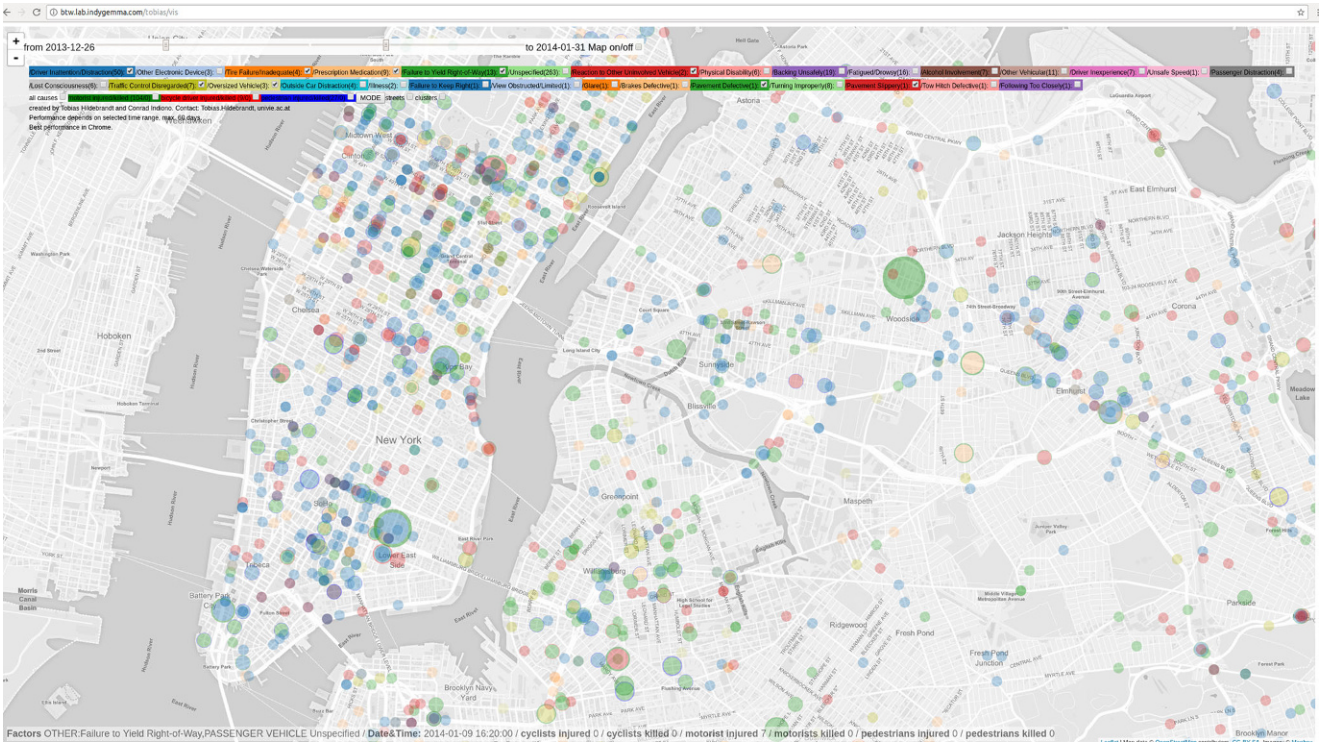

Fig. 8 Screenshot of the audiovisual user interface

- Number of injured/killed: circle size/opacity/frame thickness; Pitch range of continuous sine wave (sonification in static mode);Continuous sine wave & pitch of short acoustic notification (sonification in animated mode)
- Involved factor: fill color
- Type of injured/killed (motorist/bike/pedestrian): circle frame color

Furthermore, when clicking on an accident circle, detailed data on that accident is presented. Additionally, the most dangerous clusters and the roads with the highest number or the most dangerous accidents are highlighted on demand. The sonification itself (which at this stage is still very rudimentary) does in this case not present any data that is not at the same time also visualized, but it can be expected that the combination of these two modalities enable the user to get an overview and detect patterns faster and more efficiently, and, especially in the animated mode, get a better *feel* for the distribution of accidents over time.

As can be seen in Fig. 7, the user interface is based on JavaScript and HTML. The mapping library Leaflet (with Mapbox as mapping provider) is used in conjunction with D3[13] to visualize data. The sonification has been implemented using the Web Audio API. The source code is available on github[14], the demo can be accessed online[15], as well as video recordings[16]. More information is available on our project homepage[17].

4.5 Future Work

There are a few things we would have liked to implement, but which due to time restrictions had to be postponed. Mainly, we would have liked to display and analyze further related data sources, such as the locations of bicycle parking stations and subway exits, or times and locations of public events. Furthermore, we would have liked to integrate both user interfaces, and enhance them with further charts, graphics, and a sonification that is more sophisticated than the very basic one that has been implemented. It is planned to let the user customize which data aspects are mapped to which visual and auditory dimensions.

[13] https://d3js.org/

[14] https://github.com/TobiasHildebrandt/Audiovisual-Analytics

[15] http://btw.lab.indygemma.com/tobias/vis

[16] https://vimeo.com/user25403611

[17] http://cs.univie.ac.at/wst/research/projects/project/infproj/1096

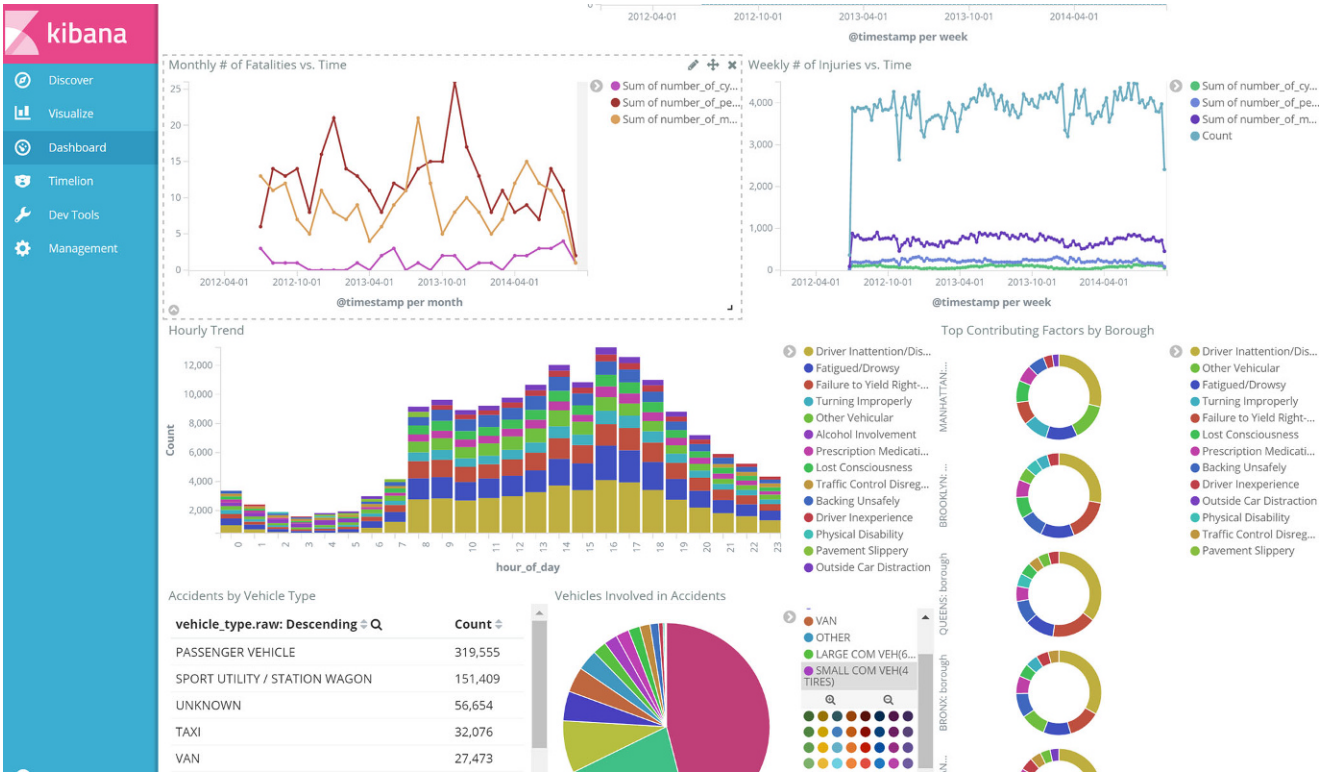

Fig. 9 Quick analysis within a simple Kibana application

5 Third Place: A Distributed Geotemporal Analysis Using Service Request & Collision Data

Martin Grimmer, Matthias Kricke, Eric Peukert

5.1 Introduction

In our first analysis of the given task, we observed that the NYC dataset is already quite well researched[18]. Many contemporary database and analysis vendors use this dataset to showcase the features of their products such as Tableau[19], IBM[20] or Elastic Search[21]. We decided to set up an analysis front-end for exploring the basic dataset of collision data. Our experience with Kibana was very positive, since the set up was very easy and the resulting visualizations were convincing as shown in Fig. 9. The tool allowed us to observe the distribution of accidents by type and the influence of a multitude factors that are given within the dataset. The "simple" analysis tasks could be solved and plotting accidents to a map of NYC can be done with the given Kibana application. Based on the initial experiences with the dataset, we designed a second more advanced analysis pipeline. Within the next subsections, we introduce our main objective, technical challenges and analysis results.

5.2 Main Objective

We developed the vision to combine existing crowd-sourced data from 311-service requests with collision data to identify regions and reasons for accidents which causes injuries or even deaths. By utilizing this model, NYC would be able to use a streaming-based application which analyzes incoming 311-service requests and prioritizes them by the probability an accident will occur. To broaden our approach for a large-scale use, we designed a distributed and scalable application which is capable of determining the probability of collisions with casualties after 311-service requests. Using this information, we can prioritize incoming 311-service requests by their relevance for government officials.

But how to correlate collisions and service requests? Our approach investigates the collisions within a radius calculated with the Haversine formula [11] and in a period after a 311-service request as it is shown in Fig. 10. A

[18] https://www.citylab.com/transportation/2014/02/mapping-new-yorks-traffic-accidents/8516/

[19] https://www.interworks.com/de/blog/modonnell/2015/08/26/exploring-nyc-vehicle-crash-data-tableau

[20] https://www.youtube.com/watch?v=KJr9dn6pmn8

[21] https://github.com/elastic/examples/tree/master/ElasticStack_nyc_traffic_accidents

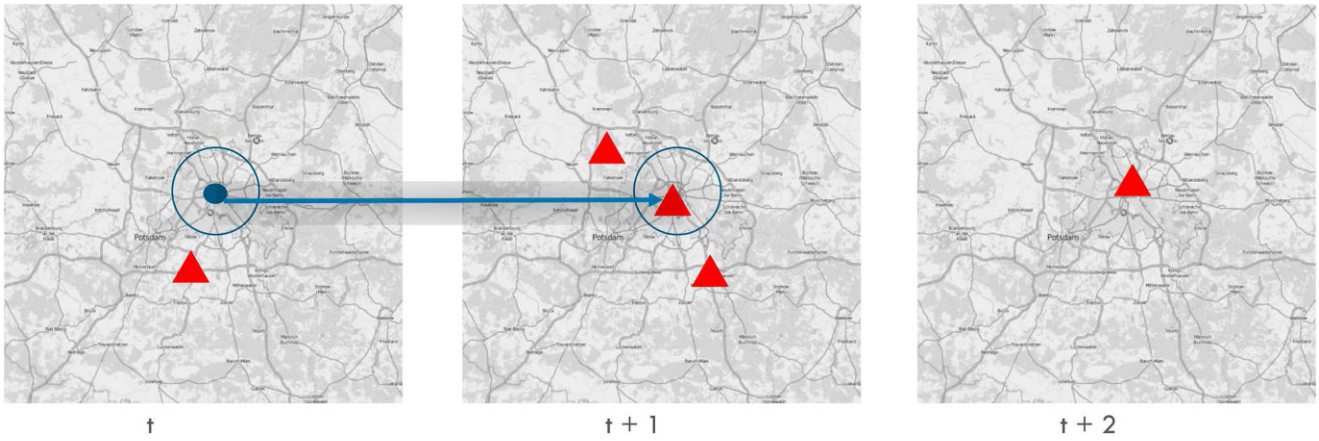

Fig. 10 The filled dot is a 311-service request where the circle is the allowed distance to a later incident which is depicted as filled triangle

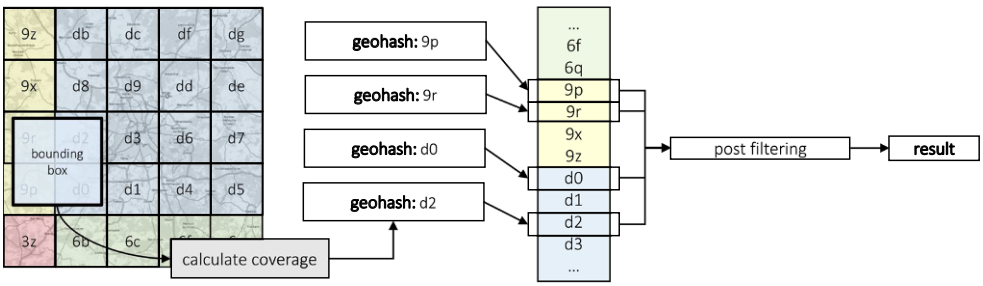

Fig. 11 Schematic view of steps needed to query the multimap index

more formal definition of this incident relation is given in Definition 1.

Definition 1 (Incident Relation φ) Be $s \in S$ a service request of the set of 311-service requests and be $c \in C$ a collision in the set of NYPD Motor Vehicle Collisions. Then the incident relation is $\varphi \subseteq S \times C$ with $\varphi(s,c) \leftrightarrow 0 \leq d_{\text{time}}(s,c) < \max_{\text{time}} \wedge \quad \text{haversine}(s_{(\text{lat,lon})}, c_{(\text{lat,lon})}) < \max_{\text{geo}}$ with $d_{\text{time}}(s,c) = c_{\text{time}} - s_{\text{time}}$.

5.3 System Architecture

For our scalable architecture and the outlook to enhance it by streaming functionality, we decided to use Apache Flink [22] for the data integration and analysis part. The visualization was done by a javascript webapp using *leaflet.js*[23].

5.4 Naive Approach

Our first approach was to implement the algorithm using Apache Flink and its DataSet API. We read the 311-service requests dataset of 311-messages and the NYPD Motor Vehicle Collisions dataset of collision messages. Afterwards we did some simple filtering to drop invalid data, e. g., without a valid timestamp or latitude and longitude coordinates. To calculate the incident relation φ (Definition 1) of the corresponding vehicle collisions and 311-service requests one has to check the time and radius condition for each possible pair of elements from sets S and C. In our first attempt, we implemented the incident relation φ with the cross operator followed by two filter operations for the time and geo-distance condition from the Flink DataSet API. This leads to $|S| \times |C|$ operations and the resulting run time was roughly 200 hours which was not practical.

5.5 Optimized Approach

To enhance the run time of our algorithm, we strongly needed to reduce the number of operations required to calculate the incident relation. We did this by replacing the cross-operation with an in-memory spatial index lookup using the *multimap* implementation from Guava[24]. In order to support fast spatial queries, we use geohashes[25] as space filling curve. In short, geohashes are a "hierarchical spatial data structure which subdivides space into buckets of grid shape". The length of a given geohash defines its precision.

[22] https://flink.apache.org

[23] http://leafletjs.com/

[24] https://github.com/google/guava

[25] https://en.wikipedia.org/wiki/Geohash

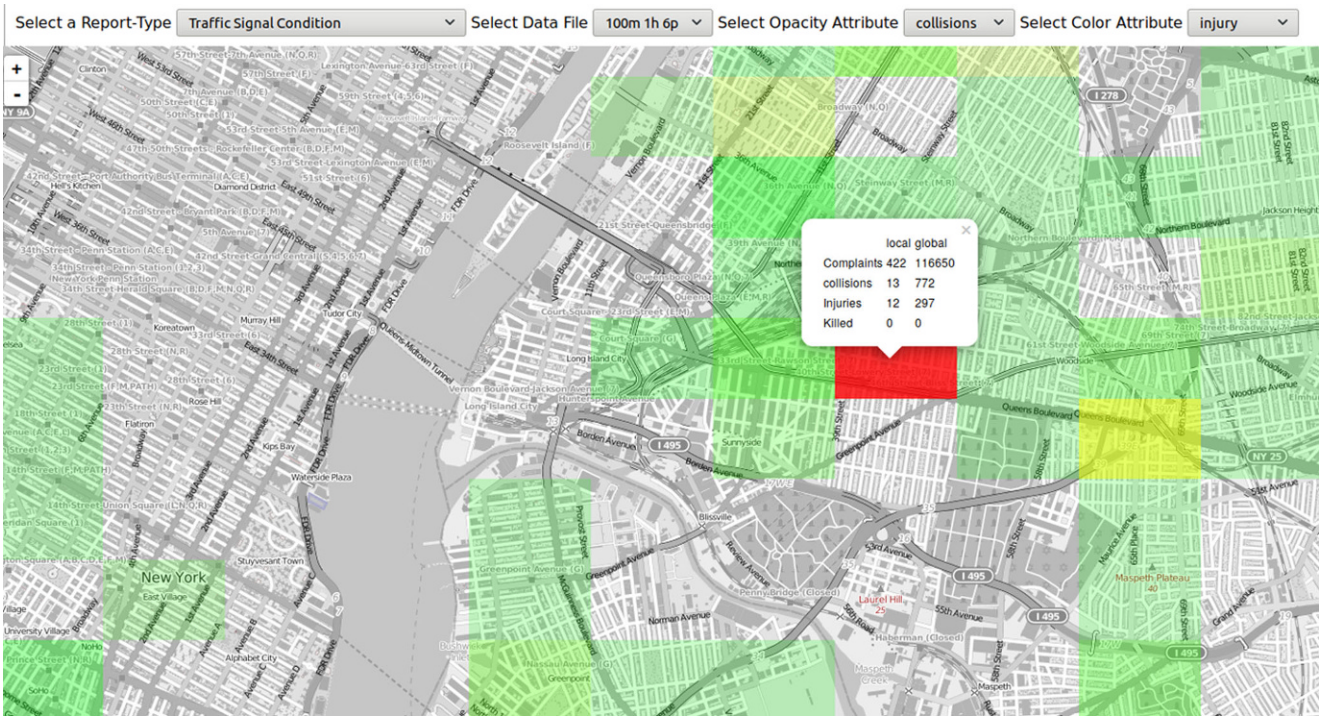

Fig. 12 Areas with a high correlation between traffic signal condition messages and injuries within 100 m and 1 h

Geohashes can be represented as *Base32* strings. For example: a geohash with length 2 like $d2$ covers an area of about $1250km \times 625km$ and a geohash of length eight (which we use) like $d2669pz2$ an area of about $38m \times 19m$. Locations which are near to each other usually have only small differences in their hashes. However, this didn't hold for all locations. There are "border" situations where hashes differ strongly from each other, even when the corresponding locations are next to each other. The calculation of geohashes for given latitude, longitude coordinates is fast with about three million GeoHash.encodeHash calls per second on an Intel i7, single thread using the Java Geohash library *geo*[26]. Figure 11 shows a map covered with (shortened) geohashes and the steps needed to query a given bounding box. During the time of the challenge, set S had about 14.6×10^6 and set C about 10^6 elements.

By using the in-memory index, we reduced the number of necessary operations from 14.6×10^{12} to roughly 17.03×10^9. This is a remarkable factor of 857 and resulted in runtimes of ~14 minutes down from ~200 hours.

5.6 Results

While experimenting with the parameters $\max_{geo}$ and $\max_{time}$ from Definition 1, we had the best results for collisions within 100 meters of the service request. In the case of $\max_{time}$, the value is different depending on the service request type. In general, one hour leads to good results.

In the interactive leaflet-based application, the computed correlations were visualized in a heat map as shown in Figs. 12 and 13. For each report type, such as ''Noise on Street/Sidewalk", the resulting correlations can be visualized. Depending on the chosen resolution of the geohash, the colored cells are smaller or bigger. A user can map computed attributes like number of collisions or injuries to the color or the opacity of a cell. In the example screenshot in Fig. 12, the number of injuries is assigned to the color attribute. The red cell indicates that there is an area where the condition of traffic signals seems to have a very strong impact onto the number of injuries within one hour after reporting a problem with traffic signals. Obviously, an operator at 311 should prioritize such reports in that area to avoid critical accidents. In a higher resolution of the geohash, the actual problematic area can be localized. The second example (cf. Fig. 13) depicts the correlation between reported noise on the street/sidewalk and the number of injuries within a timeframe of 12 hours after the report. Surprisingly, the orange area that we identified is the broadway. It should be further investigated, if larger gatherings of people initially lead to noise reports and also have an impact onto traffic security within 12 hours. Hence, in that area, reports about noise should be treated with higher priority.

In Fig. 14, we also analyzed wich report types cause the highest numbers of injuries within a short timeframe after

[26] https://github.com/davidmoten/geo/

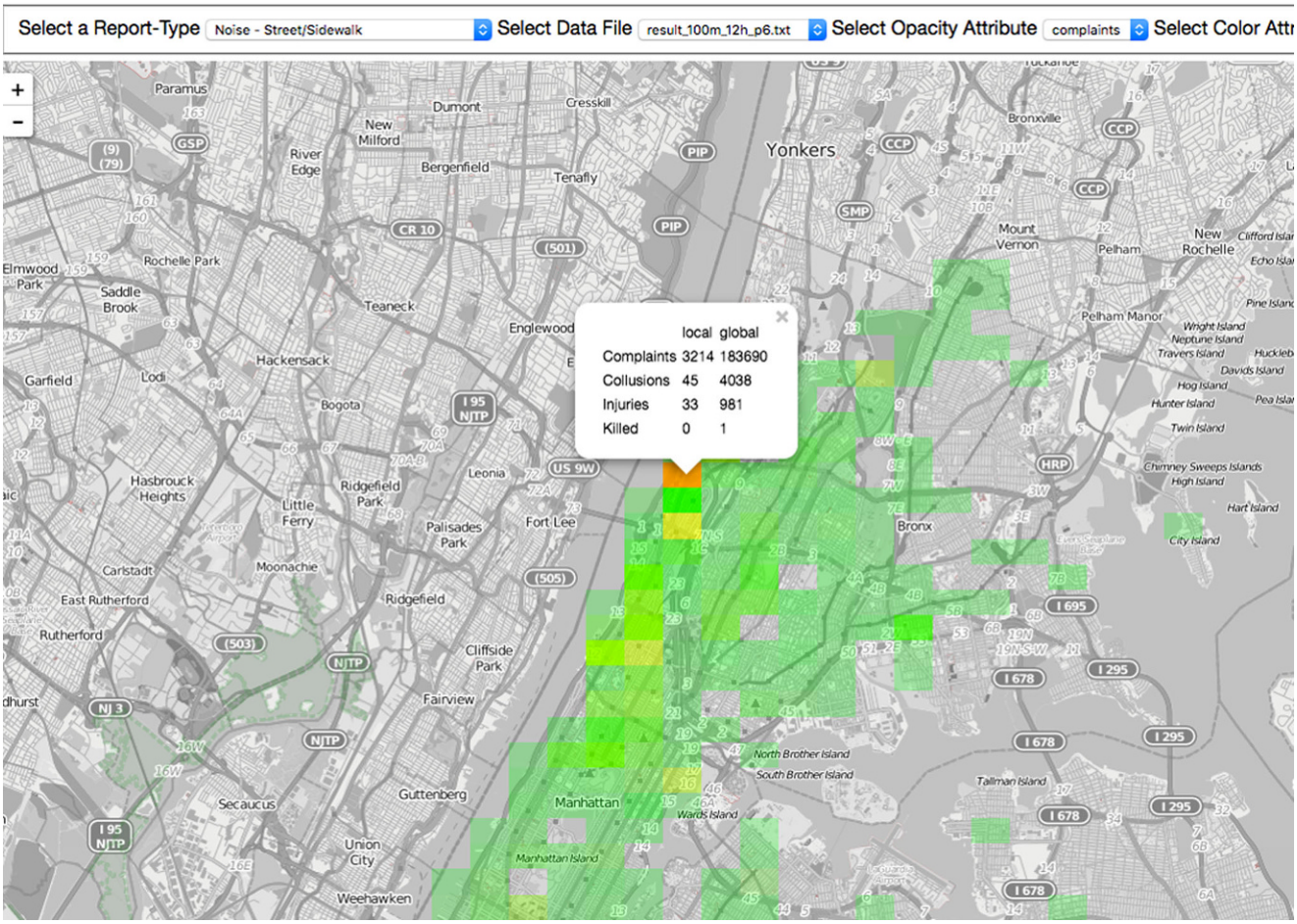

Fig. 13 Correlation between noise and the broadway

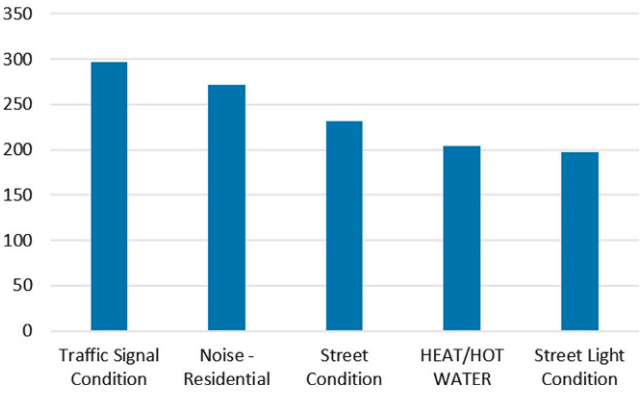

Fig. 14 The top five injury causing 311-service requests

the report. The top report types are intuitive and an explanation is easy. However for HEAT/HOT WATER, we initially did not see a reason why this might have a direct impact on accidents. Within the presentation of the results at the BTW, a jury member could give a reason. The service personnel of the public water companies often park in second row. These vehicles might create unknown obstacles that force car drivers or cyclists to change lanes. And this might then cause the measured increase in severe accidents. In Fig. 15, we could also show that the distribution of report types and number of caused injuries follow the pareto principle also known as the 80/20 rule. 20 percent of the reports cause more than 80 percent of injuries. Hence, these report types should be treated with high priority.

The given examples show that the identified temporal correlation between 311-reports and immediate accidents is valuable. With the help of the implemented analysis pipeline and visual analysis tool, many of such correlations can be identified. Many of them should lead to changes in action at 311. The presented approach is scalable and could be rolled out to the whole US and other countries given that the required data is collected in the first place. We plan to publish the resulting application at the ScaDS-Blog.

Acknowledgements The organizers of the Data Science Challenge thank the participants and the jury for the time they invested. Furthermore, we would like to thank INFOS for sponsoring the price money of the event.

This work was partially funded by the IST-Hochschule University of Applied Sciences and by the PhD program*Online Partici-*

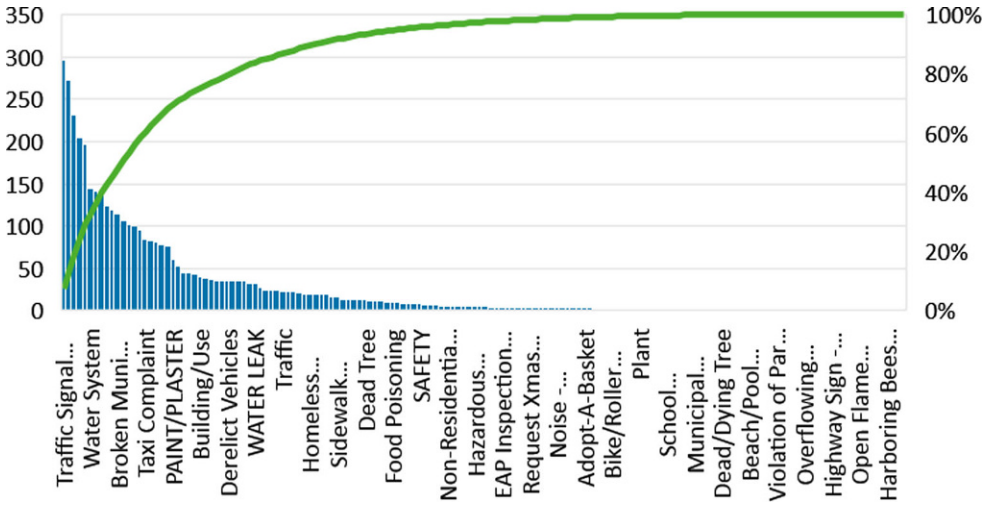

Fig. 15 20% of the events are resulting in 80% of the injuries

pation, supported by the North Rhine-Westphalian funding scheme *Fortschrittskollegs*.

This work was partly funded by the German Federal Ministry of Education and Research within the project Competence Center for Scalable Data Services and Solutions (ScaDS) Dresden/Leipzig (BMBF 01IS14014B) and Explicit Privacy-Preserving Host Intrusion Detection System EXPLOIDS (BMBF 16KIS0522K).

References

1. Australian Bureau of Statistics (2017) Time series analysis: the basics
2. Chaudhuri S, Dayal U (1997) An overview of data warehousing and olap technology. ACM Sigmod Rec 26(1):65–74
3. Cortes C, Vapnik V (1995) Support-Vector Networks. Mach Learn 20(3):273–297
4. Ho T (1995) Random decision forests. In: Proceedings of the 3rd International Conference on Document Analysis and Recognition, pp 278–282
5. Keim D, Andrienko G, Fekete JD, Görg C, Kohlhammer J, Melançon G (2008) Visual analytics: definition, process, and challenges. In: Information visualization. Springer, Berlin Heidelberg, pp 154–175
6. Kisilevich S, Mansmann F, Nanni M, Rinzivillo S (2010) Spatio-temporal clustering. Springer, Boston, pp 855–874
7. Maciejewski R, Rudolph S, Hafen R, Abusalah A, Yakout M, Ouzzani M, Cleveland WS, Grannis S, Ebert DS (2010) A visual analytics approach to understanding spatiotemporal hotspots. IEEE Trans Vis Comput Graph 16(2):205–220
8. NYC Taxi & Limousine Commision (2017) Tlc trip record data
9. Slocum TA, McMaster RB, Kessler FC, Howard HH (2005) Thematic cartography and geographic visualization. geographic information science. Pearson, Prentice Hall
10. Thomas J, Kielman J (2009) Challenges for visual analytics. Inf Vis 8(4):309–314
11. Van Brummelen G (2013) Heavenly mathematics: the forgotten art of spherical trigonometry. Princeton University Press, Princeton
12. World Health Organization (2017) Top 10 causes of death