Outlier Detection in Bus Routes of Rio de Janeiro

Aline Bessa, Fernando de Mesentier Silva, Rodrigo Frassetto Nogueira

Abstract—Recently, the city of Rio de Janeiro released the real-time GPS coordinates of all public buses that operate in the city. The system generates almost 1 million GPS entries per day, and a reasonable amount of them are outliers, that is, trajectories that do not follow the expected behavior when compared with the majority. In this project, we developed visualizations that help users identify and understand the behavior of outlier buses (outliers), which were automatically detected by a Convolutional Neural Network (CNN) especially designed for this application.

1 INTRODUCTION

Recently, the city of Rio de Janeiro released the real-time GPS coordinates of all public buses that operate in the city. This data allows broad range of monitoring and analysis of the public transportation system. One possible application is the automatic detection of events, such as accidents, blocked roads, traffic jams and broken buses.

In this project, we developed a visualization tool that help users identify and understand outlier buses (outliers) in the bus routes of Rio de Janeiro that we identified in by a machine learning algorithm. These outliers can be spatial – e.g. buses running outside their average route (far from where they are likely to be) – and temporal – e.g. delayed buses. To understand the nature of outliers better, let us start off by focusing on spatial outliers.

Spatial Outliers In Rio, buses travel in specific routes according to their lines and are required to provide their service in a timely manner. Unfortunately, buses may alter their services for a number of reasons such as roadblocks (due to unexpected happenings, following holidays celebrations etc) and misbehaviour by the bus drivers (it is not uncommon for bus drivers of the same line to race each other during their runs). Also, GPS data captures are active when buses are moving to their garage and when they are stopped (either by malfunction or at the garage, outside their service hours). In these occasions, these buses are behaving as spatial outliers, and thus should be detected. The detection of spatial outliers for each bus line helps to answer the following questions: (i) What bus lines have more spatial outliers? (ii) How do these outliers stray away from their routes, and where? (iii) In which days are there more spatial outliers for a certain line? (iv) What buses are following routes that are so unexpected that they are probably incorrectly labeled (i.e. their bus line ids in the dataset are probably wrong)?

Temporal Outliers As for temporal outliers, they stand for buses that are taking too little or too much time to go from one bus stop to the next, when compared to the average time taken by other buses in the same line. More specifically, a bus behaving as a temporal outlier emits a sequence of GPS captures that show an unusual average speed when compared to other GPS sequences that have similar characteristics (same line, similar date etc). Buses can behave as temporal outliers for several reasons: traffic jams, malfunctions, direction errors committed by the bus drivers etc. It may also be the case that a specific bus is emitting a GPS data entry with a wrong timestamp, giving the impression that it is more delayed or more on time than it actually is. Finally, there is a correlation between being a spatial outlier and being a temporal one, as some of the buses who strayed from their routes will usually take more time to hit their next stops. In spite of it, in this work we will treat temporal and spatial outliers as separately as possible, as an attempt to end up with clearer visualizations. The detection of temporal outliers for each bus line helps to answer the following questions: (i) What bus lines have more temporal outliers? (ii) Are these temporal outliers usually running faster or slower than the average speed? (iii) In what dates, or hours, do these lines get more affected by temporal outliers? (iv) Are there any known reasons to back up the existence of specific temporal outliers?

There are different ways to detect outliers. One can argue that just by visualizing the buses (on a map, for example), it is possible to detect which of them are outliers. From our experience, it is not so simple to proceed like this: the Rio bus system generates almost 1 million GPS entries per day, and a reasonable amount of them are outliers. To detect several outliers from different bus lines in a large-scale fashion just by looking at them is very time-consuming, to say the least. Finally, and most importantly, if the users have to identify the outliers manually, it takes more time for them, by interacting with the visualizations, to come up with answers for the questions posed in the paragraphs above.

This created the need of coming up with a model to detect outliers automatically. Afterwards, the detected outliers are exposed to the end-users throughout visualizations, which comprise the core of this project. In this way, end-users can interact with the visualizations in a more effective way, coming up with answers to the above mentioned questions faster. We analyzed some different models for outlier detection: KNN classifiers, clustering methods, and Convolutional Neural Networks (CNNs) [1]. CNNs were the best studied option, both in terms of performance, as the data we use is large, and in terms of detection accuracy. As the study of these models is a bit out of the scope of this project, we will only discuss our use of CNNs tangentially, in the last report of this course.

With the CNNs, we learn a model that detects temporal and spatial outliers in a dataset comprised by bus routes, described in Section 3. An initial challenge in this project is to make sure that the model detects valid outliers, and in order to do it visualizations are very useful. It is hard, for example, to understand whether a bus is away from its route without a geospatial visualization. As validating the CNN model in a very thorough manner is out of the scope of this project, the team member Fernando, who was born and raised in Rio, sampled some of the CNN outputs and analyzed them visually by plotting them in a map. His conclusion was that the CNNs seemed to be quite accurate and that they were behaving better than our previous studied options. We intend to evaluate our CNNs in a more principled way as a future work, also by using some visualizations to inspect their accuracy.

1.1 Description of End-Users

After having an automatic outlier detection model, we can provide a tool with which end-users can identify and analyze outlier buses. These end-users can be either urban data analysts/traffic engineers monitoring Rios bus traffic or regular citizens who wish to learn more about unusual traffic behavior or bus service. We intend to create visualizations that will allow these end-users to come up with answers to the following questions: (i) What bus lines have more outliers? (ii) Do lines with more outliers usually have more captured GPS points? (iii) In which days are there more outliers for a certain line? (iv) What bus lines have more spatial outliers? (v) How do these outliers stray away from their routes, and where? (vi) In which days are there more spatial outliers for a certain line? (vii) What buses are following routes that are so unexpected that they are probably incorrectly labeled (i.e. their bus line ids in the dataset are probably wrong)? (viii) What bus lines have more temporal outliers? (ix) Given a temporal outlier, is it running faster or slower than its expected average speed? (x) In what dates, or hours, do these lines get more affected by temporal outliers?

Currently we have access to regular Rio citizens who are willing to use our visualizations to better understand Rios outlier buses. We are also in contact with data analysts and traffic engineers who work in the Secretaria de Urbanismo of Rio (Rios urban office) to help us test our prototypes.

1.2 First Steps and Goals

This project is a follow-up of a study conducted in the Massive Data Analysis course, during the Fall of 2014. By that time, we developed some techniques to identify outlier buses in Rio de Janeiro, such as KNN classifiers and clustering methods, but they did not scale well. Nevertheless, we have recently developed an application of CNNs that scales, and this is the starting point of this project. The goals of this project are twofold: (i) to provide visualization tools that indicate the detected outliers, and (ii) to show that outlier detection can help understand more about Rio de Janeiro and its primary source of public transportation. More specifically, we will implement a tool to visualize temporal and spatial outliers. Additionally, we will have meta visualizations for the bus lines, so the user can start off by deciding which line she wants to investigate and then accessing its temporal and spatial visualizations. This suggestion was given by Professor Enrico as a means of being able to investigate every bus line without having to access too much data at once.

2 RELATED WORK

In this section, we present some papers and news articles related to our project.

Visualizing the Behavior of Higher Dimensional Dynamical Systems [2] - In this paper, the authors deal with the visualization of trajectories of high-dimensional dynamical systems which form a Lnndata set of a smooth n-dimensional flow. Three methods that are based on the idea of parallel coordinates are presented and discussed. Visualizations done with these new methods are shown and an interactive visualization tool for the exploration of high-dimensional dynamical systems is proposed. Although the resulting tool is used in many different applications, it does not fit for visualization of large amounts of samples (which is our case) and it does not support dynamic selection of time ranges via user interface, which we believe to be a necessary functionality for the visualization of outliers.

Visualizing and Understanding Convolutional Networks [3] - In this paper, the authors introduce novel visualization techniques to explore CNNs in order to understand why they perform so well, or how they might be improved. The technique gives insight into the function of different layers of the CNNs and the operation of the classifier. Through visualizations, they find model architectures that outperform state-of-the-art ones and also discover the performance contribution from different model layers. In the first part of our project we also use visualization – in particular, the visualization of outliers detected by our CNNs – to refine and improve our model. Our approach is more indirect than theirs, as we do not visualize the actual layers of the CNNs and their contributions to the outlier detection.

Vistradas: Visual Analytics for Urban Trajectory Data [4] - In this paper the authors present a system, named Vistradas, for visual analytics of urban trajectory data. Vistradas allows users to perform analysis of bus uniformity, verification of bus route, and the impact of events in bus traffic. This paper is our closest related work, as we use the same datasets and have derived our initial tasks last Fall from it. Nonetheless, they do not focus on outlier detection, which is the main goal of our project.

Sao Paulo tests software to modernize bus management - According to this article, the Traffic Bureau of Sao Paulo (SPTrans) is testing a software to modernize and automate its bus management. The software, created by an american company named Urban Engines, gathers information from bus tickets, bought by the passengers, and from the buses GPS. This system allows the bureau, through visualizations on a map, to see in real-time where the buses are, their average speed, and how many passengers get in and off the buses in each bus stop. This work at SPTrans is related to our project because it is an initiative to analyze bus traffic in Brazil by using visualizations. Nonetheless, it is not clear if, in this system, it is possible to visualize possible outliers in a straightforward way, and whether or not the system uses models over historical to learn traffic patterns. It is very likely that the tasks we address in our project will yield different visualizations and solve different problems, in comparison with this work.

Visual Exploration of Big Spatio-Temporal Urban Data: A Study of New York City Taxi Trips [5] - In this paper the authors present TaxiVis, a system for data-driven analysis of taxi trips in New York City. The taxi data contains the geographical and temporal information associated with each trip, and the model shown allows for users to perform visual queries. Although the work does not mention approaches for outlier detection, the interaction with the visualization of geospatial-temporal data is a source of inspiration to some of the tasks we wish to accomplish.

Designing Flow Visualizations for Oceanography and Meteorology using Interactive Design Space Hill Climbing [6] - In this paper authors show a 2D flow visualization for Oceanography and Meteorology. The head-to-tail arrow representation used to show motion in this work is an inspiration to the visual cue we aim for with the continuous spatial transitions of buses throughout time as it encodes bus movement direction in each time frame.

Stacking-Based Visualization of Trajectory Attribute Data [7] - In this paper the authors present a visualization for trajectories. Through a hybrid 2D/3D display trajectory bands are stacked on top of a map that is shown for space context. This approach provides a solution for space and time values being displayed at the same time. The visualizations we propose cover some of the same aspects they were tackling, but we felt they were a better approach to our problem since we want the outlier analysis to be our focus.

3 DATA ANALYSIS AND ABSTRACTION

In this project we make use of two datasets released by data.rio. The first dataset corresponds to GPS entries for buses in transit in Rio de Janeiro, and the second one corresponds to GTFS data, which contains general information about the bus routes,

such as bus stop locations and expected schedules. Hereafter we will refer to the first dataset as the **bus** dataset and to the second one as the **GTFS** dataset. These two datasets are from the type table.

The entries in the bus dataset are generated by a live stream in the data.rios website. Our data was obtained from September 26, 2013 to January 9, 2014. There are more than 9,000 buses for approximately 490 bus lines in the city of Rio de Janeiro. In total, this dataset is comprised of 151,730,254 entries, with approximately 23GB. Initially, each GPS data entry has the following attributes:

- the bus ID, which is unique for each vehicle (categorical)
- the point ID, which is unique for each point in Rios surface where a GPS entry was captured (categorical)
- the bus route ID, that corresponds to the bus line (categorical)
- the bus name, which seems to indicate the bus type, although we need to check it out (categorical)
- the hour when it was collected, from 00 to 23 (quantitative)
- the day when it was collected, from 1 to 31 (quantitative)
- the month when it was collected, from 1 to 12 (quantitative)
- the year when it was collected, either 2013 or 2014 (quantitative)
- the bus speed when it was collected, in km/h (quantitative)
- a corresponding latitude (quantitative)
- a corresponding longitude (quantitative)
- a timestamp in UTC format – the day, month, year, and hour in this timestamp are equal to the values in the corresponding fields (quantitative)
- a description of the entry, with its corresponding speed and timestamp (quantitative)

3.1 Training a CNN over the bus dataset

In order to train a CNN using this dataset we transformed the dataset in the following way: each input sample has a sequence of 20 GPS coordinates (that corresponds to approx. 2 hours) of a particular bus id and route. Also, each sample has 3 channels: longitude, latitude and timestamp. The network is trained in a supervised fashion and predicts a route for each sample. Thus, since there are 486 distinct routes/lines in the Rio de Janeiro Bus System, the network outputs the probability for 486 classes. We used the following architecture:

- 3 convolutional layers with filter size of 3 + pooling layer with kernel size and stride of 2.
- 3 convolutional layers with filter size of 3 + pooling layer with kernel size and stride of 2.
- 3 convolutional layers with filter size of 3 + pooling layer with kernel size and stride of 2.
- 2 Fully connected layers with 4000 neurons each.
- 1 Fully connected layer with 486 neurons.

We separated 10% of the dataset for model validation. The learning rate is chosen during the validation phase and the best value found was 10-5. Stochastic Gradient Descent with batch size of 50 was used and the training stops after 2 million iterations (which takes approx. 2 days). We used Caffe framework and the GPUs nodes from NYU HPC cluster. Currently, our best model has an accuracy of 94.9%.

After the model is trained, we predict the outliers by inputting to the model a sequence of 20 consecutive GPS entries for a same bus and line. The model outputs two fields:

- predicted bus route ID (categorical)
- confidence percentage for the predicted bus route ID (ordinal)

The input sequence is considered an outlier if the confidence percentage is below a certain threshold (10% in our preliminary experiments) or the predicted bus route ID is different from the one assigned to the sequence. A sequence of GPU entries is needed because it would be hard (if not impossible for temporal outliers) to detect an outlier using only one GPS entry.

3.2 The GTFS dataset

In general each route consists, usually, of two trips, from the final bus station A to the final bus station B and back. The GTFS dataset contains a complete definition of each such trip as a sequence of line segments tracing the streets of the route from origin to destination. In total, this dataset is comprised of 62,032 entries. Each entry in the GTFS dataset is comprised by the following attributes:

- a bus route ID corresponding to a bus line (categorical)
- a description that corresponds to the full name of the bus line (categorical)
- an agency, that corresponds to the company responsible for operating that line (categorical)
- a sequence number that indicates where that entry is in the order of the bus lines stops, from the beginning to the end of its
- route (ordinal)
- a corresponding latitude (quantitative)
- a corresponding longitude (quantitative)

We are not currently using the GTFS dataset, but given that it has all of Rios official bus stops, it can be interpreted as a source of official routes for each line. It is likely that this type of information will help us curate our CNNs model, and this is something that we may explore in the future.

4 VISUALIZATION AND INTERACTION DESIGN

In this section we present our current design and comment on its implementation. Our design seeks to follow the Shneidermans mantra (overview first, zoom and filter, then details-on-demand) so we can deal with the large amount of information we have in a more organized fashion. Our visualization has multiple views, all of which the user can interact with in order to get detailed information about the bus outliers. In Figure 1 we show all views from our visualization.

Our visualization is composed of 4 different views:

1. The top view is a bar chart highlighting the calendar days to which we have data on outliers;
2. The middle left view is a bar chart for each bus line, indicating its number of detected outliers for a specific day;

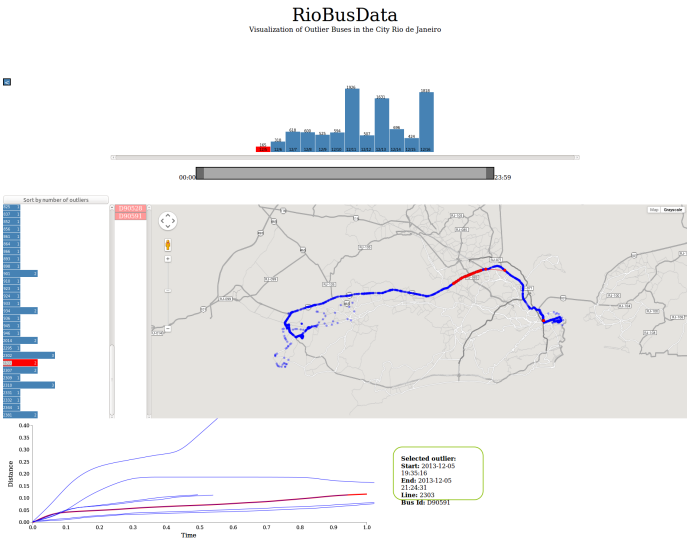

Fig. 1. The overview of our visualization tool.

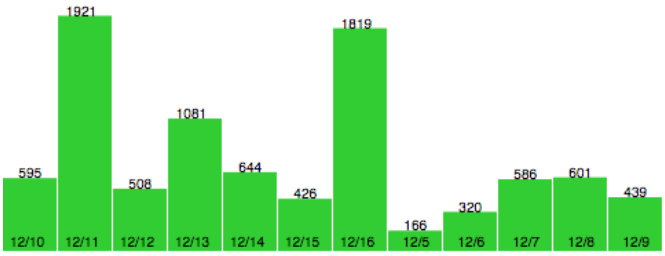

Fig. 2. Detailed view of the Calendar Bar Chart.

3. The middle right view is a map where regular and outlier bus data points are plotted;

4. The bottom view is a temporal chart where the user can see how the distance of a certain bus to the end of its route evolves over time;

We found that these four views are the necessary and sufficient elements for a simple but effective visualization of outliers in the public bus transportation system. The calendar and line bar charts inform the users about the distribution of outliers by date and lines while simultaneously give them the ability to select the information to be displayed. The map view plays a major role in helping to understand the spatially distribution of outliers while the temporal chart helps to understand the temporal outliers.

4.1 Calendar bar chart

Figure 2 shows a bar chart for some calendar days. The position on the X axis encodes the available days and the value on the Y axis encodes how many outliers there are in each day. By clicking on a bar the user selects that day and the data for the other views are filtered according to it. We will add labels to the axes and a functionality that allows the user select as many days as she wants soon.

The marks in Figure 2 are bars that correspond to the amount of outliers for each day. The table below synthesizes the attributes mapped on this visualization.

Attribute	Visual Channel
Day	Position X (Categorical)
Number of outliers	Position Y (Quantitative)

Table 1. Attributes and channels for Calendar bar chart.

4.2 Line-Outliers Bar Chart

In Figure 3, we have a zoom on the line outlier bar chart, on the left in Figure 1. After a day is selected in the calendar bar chart (or, in the future, any given number of days), the line outlier bar chart can be used to select one or more lines to be displayed on the map in Figure 1. Besides this filtering functionality, this bar chart encodes the number of detected outliers in each line for a given day. We may add a title or some sort of hint to make it clear that the numbers on the left are the line IDs and the numbers on the right are the number of detected outliers.

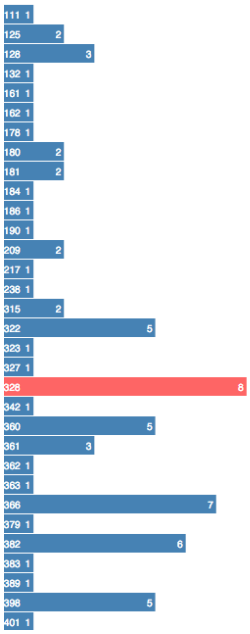

Fig. 3. Bar chart visualization for number of outliers per line in a given day.

The marks in Figure 3 are bars that correspond to the amount of outliers for each line. The table below synthesizes the attributes mapped on this visualization.

4.3 Map View

The map view is the core of our work. It is where the most relevant piece of information is outlined: regular buses in blue and detected outliers in red. The plotted data corresponds to a day and a bus line – or, alternatively in the future, to a range of days and range of bus lines. Due to the large number of points available we had to sample them (this amount is configurable - the default is 1000).

Outliers in the map visualization that are far from blue points are spatial outliers. They are not following their routes adequately, following the pattern delineated by regular buses. When outliers are mixed with blue points they were usually identified as such because they are temporal outliers. In other words, they are following their routes correctly, but there is something wrong with their speeds (they are either below or above average when compared with other buses with the same

Attribute	Visual Channel
Line	Position Y (Categorical)
Number of outliers	Position X (Quantitative)

Table 2. Attributes and channels for the Line Outlier Bar Chart.

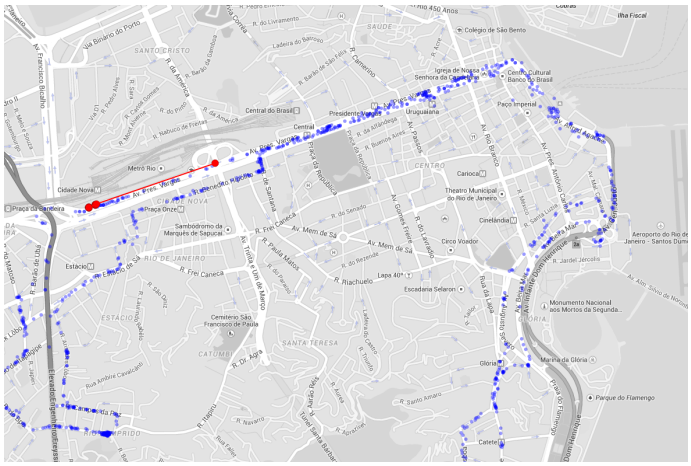

Fig. 4. The map view with regular buses in blue and outliers in red.

Attribute	Visual Channel
Latitude (transformed into a 2D value)	Position X (Quantitative)
Longitude (transformed into a 2D value)	Position Y (Quantitative)
A boolean value indicating whether or not the point belongs to a detected outlier segment (i.e. is an outlier)	Color (Categorical)

Table 3. Attributes and channels for the Map View.

line ID and similar timestamps). It is very straightforward to verify spatial outliers in the map view, but to better analyse temporal ones, another visualization was necessary, as explained in Section 4.4.

For the map view, the spatial layout technique is a map with plotted points in their GPS coordinates. The marks are points that correspond to buses moving across Rio de Janeiro. There are labels attached to these points indicating their bus IDs. The table below synthesizes the attributes mapped on this visualization.

4.4 Temporal Chart

The temporal chart is useful to inspect the behavior of specific outliers, in order to check how close or far they get to a bus stop throughout time. It works this way: in the map view, the user selects an outlier she wants to inspect. The X axis of the temporal chart will represent the time, ranging from the moment the outlier was detected until it reaches a specific bus stop (cycle 0), from the moment it reaches this bus stop until it reaches it again (cycle 1) and so on. The Y axis represents the distance from this outlier to this specific bus stop. Some regular (non-outlier) buses are also plotted for comparisons sake. Only regular buses that fall within the same interval of the outlier are plotted. This constraints the user to compare outliers and regular points within the same time of the day. Right now we are still working on the computation of cycles for buses in different lines, so the user can click on an outlier in the map view and check how it behaves for different cycles in that day. In a sense, a cycle can be seen as an entire trip of the bus. Consequently, this functionality is not yet available in the current github demo.

For the temporal chart, the spatial layout technique is a chart with lines generated by the interpolation of the GPS entries we have for the buses. The marks are lines that correspond

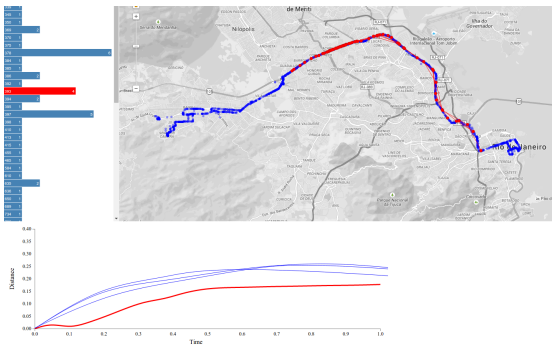

Fig. 5. Temporal chart to analyze outlier buses. The X axis corresponds to the amount of time the bus is in a certain cycle of its trajectory, and the Y axis corresponds to its distance to the first point of the outlier.

Attribute	Visual Channel
Current time (it ranges from the moment the outlier starts a cycle to the moment it ends it)	Position X (Quantitative)
Current distance (it is the distance from the bus to the bus stop that corresponds to the end of its cycle)	Position Y (Quantitative)
A boolean value indicating whether or not the point is an outlier	Color (Categorical)

Table 4. Attributes and channels for the Temporal Chart.

to buses moving across Rio de Janeiro to a specific bus stop of its line, forming cycles. There will be labels attached to these points indicating their bus IDs. The table below synthesizes the attributes mapped on this visualization.

4.5 Filter by Time

Another implementation that is important to mention is that we added two sliding controls that filter the points by start time and time range (window size). Figure 6 shows an example where the start time and end time are set to 8:17 AM and 9:03 PM, respectively. In this case, only points whose timestamps are between this range will be visible. With this functionality, the user can yet filter more information, by focusing exclusively on outliers and regular points that occurred in a particular time of the day. This helps come up with answers for the questions above in a finer-grained fashion.

4.6 Additional Implementations

We modified the underlying map to be displayed in grayscale (Figure 8). This made the visualization of outliers much clearer

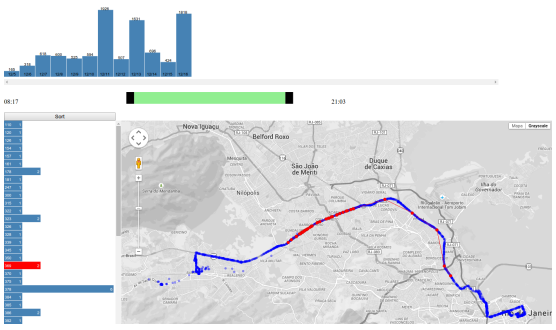

Fig. 6. Sliding controls to filter information by time out from the map visualization.

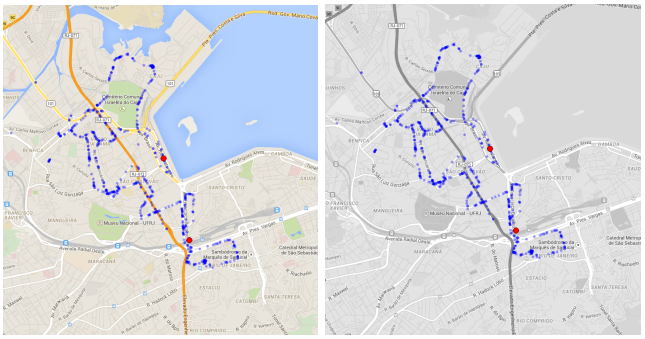

Fig. 7. Left: Map in color; Right: Map in grayscale to avoid distraction by elements such as roads and lakes.

as some distracting elements such as roads, lakes and other map signs are not highlighted by the color anymore.

5 TASK ANALYSIS AND QUESTIONS

In this section we explain how our interface, comprised by the visualizations described from the previous section help the user come up with answers to the questions posed in Section 1.1:

1. What bus lines have more outliers? Right now, given a day, it is possible to answer to this question just by looking at the Line Outlier Chart on the left (see Section 4.2). In the near future, when all days can be selected, the Line Outlier Chart will enumerate the number of outliers for all lines for all days, leading to a more complete answer to this question.

2. In which days are there more outliers for a certain line? By selecting different days in the Calendar Bar Chart and checking the corresponding number of outliers for a certain line in the Line Outlier Chart, the user can compare different days and come up with an answer to this question.

3. What bus lines have more spatial outliers? By selecting different lines in the Line Bar Chart and looking at the map, it is possible to identify their spatial outliers (red points that are distant from blue points). As a consequence, it is possible to compare lines with these behaviors and come up with an answer to this question.

4. How do these outliers stray away from their routes, and where? - This analysis comes naturally from the Map View, as it outlines regular and outlier buses on the actual streets, making it easy to see how the outliers are different from the regular buses.

5. In which days are there more spatial outliers for a certain line? By selecting different days in the Calendar Bar Chart, the user can then look at the Map View and check how the amount of spatial outliers, represented as red points far from regular trajectories, varies.

6. What buses are following routes that are so unexpected that they are probably incorrectly labelled (i.e. their bus line ids in the dataset are probably wrong)? - This analysis also comes naturally from the Map View, by detecting buses that are clearly running over different streets most of the time.

7. What bus lines have more temporal outliers? By choosing different lines in the Line Outlier Chart, the user can observe the amount of outliers mixed with the blue points, which are temporal outliers, for each line. The user can then compare the amount of temporal outliers for different lines.

8. Given an outlier, is it running faster or slower than its expected average speed? By clicking on an outlier in the Map View, its cycles throughout the day will be plotted in the temporal chart, alongside other regular buses from the same line. This will allow the user to make speed comparisons between average buses and outliers, helping come up with an answer for this question.

9. In what dates, or hours, do these lines get more affected by temporal outliers? By picking different dates in the Calendar Bar Chart, maybe filtering by a specific range of hours, and selecting always the same line in the Line Outlier Chart, the user can compare the amount of temporal outliers for different dates. The key here, which we should probably highlight in a helper for the visualization, is to focus on red points mixed with the blue ones.

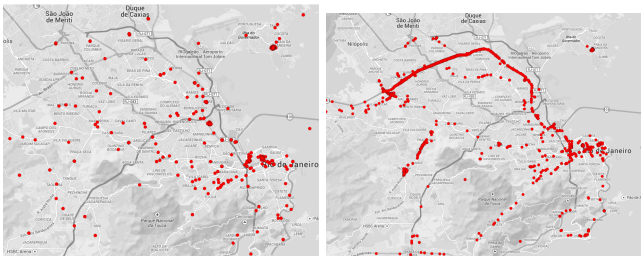

Fig. 8. Left: All outliers plotted on a regular day. Right: Multiple outliers detected on some of the main avenues of the city, which indicates a possible traffic jam.

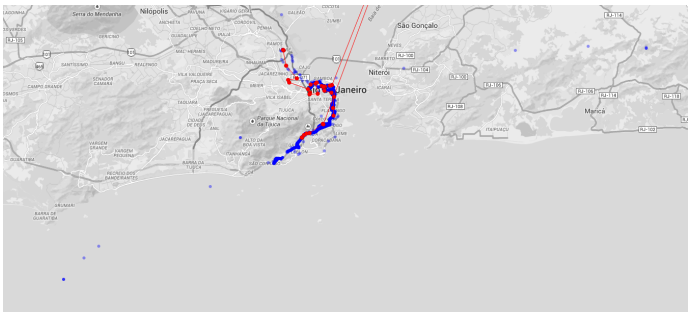

Fig. 9. False Negatives: points in the ocean and outside the city bounds that were not marked as outliers.

6 FINDINGS AND INSIGHTS

In this section we provide some interesting findings and insights we had thanks to the visualization tool.

6.1 Finding 1: Traffic Jams

The outlier detection algorithm was able to detect traffic jams even without being explicitly trained to do so. Figure 8 shows the view of all outliers for a regular day (left image) and for day with congested roads (right image).

6.2 Finding 2: False Negatives

The visualization tool also helped us to find false negatives, that is, points marked are regular but that should be marked as outliers. As an example, Figureẽffig:falsenegatives shows some (blue) points in the ocean and outside the city bounds that were not marked as outliers.

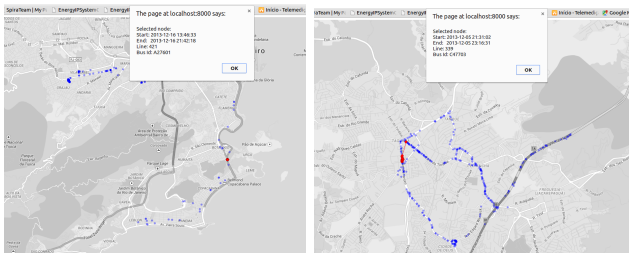

Fig. 10. Two examples of buses that stood for many hours in some streets of the city.

Fig. 11. Noise GPS entries detected by the outlier detection algorithm.

6.3 Finding 3: Possible Broken Buses

The outlier detection algorithm was able to automatically detect possible broken buses. The Figure 10 shows buses that stood over for more than one hour in busy streets of the city.

6.4 Finding 4: GPS Noise

The algorithm was able to detect GPS noise as outliers. Figure 11 shows two examples that the buses are following their normal route but they suddenly appears many miles far from the city and comes back in the next moment. These examples inspired us the idea that the outlier detection algorithm could also be used as pre-processing tool to remove noisy samples from the dataset.

6.5 Finding 5: A bug found in the Machine Learning code

We implemented a functionality that allows the visualization of a sequence of outliers by connecting consecutive points with a line, as seen in Figure 12. Unexpectedly, this functionality helped us to find a bug in the algorithm that converts the raw data to the format used by our machine learning model. More specifically, we wrongly assumed that the raw data was sorted by time, but it turned out that some of points (less than 1%) are incorrectly placed in between the sorted raw data. Since our machine learning algorithm needs sequences of points that are consecutive in time, we had to insert a sorting phase in the conversion algorithm. After this correction, the system predicts much more plausible outliers.

7 LIMITATIONS AND FUTURE WORK

Currently, our tool does not have the following functionalities that might be useful to understand outliers:

- Multiple days visualization: Currently, only one day can be visualized at once. Multiple days will be useful to understand how the outliers change over time (for example, for different days of the week).
- Multiple line selection: It might be interesting to compare multiple lines and check if there are intersections of outliers that may indicate a blocked road, for example.
- The system only compares outliers with regular points of the same day. The dates of the raw points should be different of the outlier being analyzed since the user might want to know the regular behavior of buses through a larger number of days, not just the day the outlier occurred. Another reason is that for a given day the behavior of buses might abnormal for all buses, not only the outlier. A third reason is that the user might want to compare the outlier with the regular points the system learned. In our view, this is the major limitation of our current implementation.

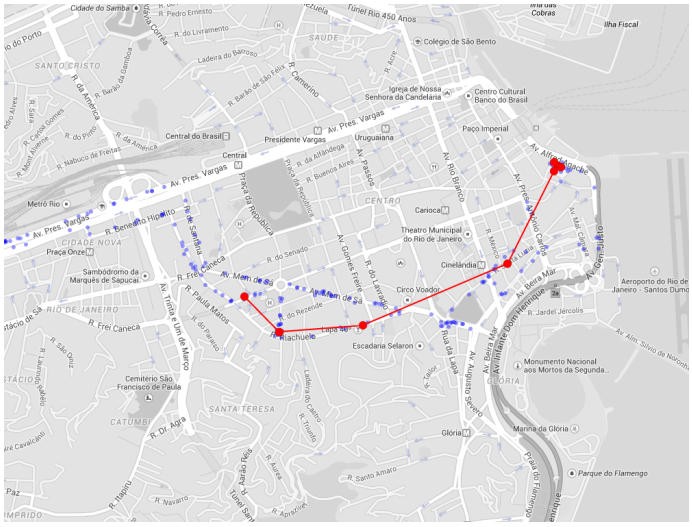

Fig. 12. Consecutive points of the same outlier are connected by a line segment.

8 CONCLUSIONS AND LESSONS LEARNED

In this work we presented a visualization tool that help the user to understand and analyze outlier buses in the city of Rio de Janeiro that were detected by a machine learning algorithm. Thanks to this tool, we could better evaluate the algorithm, find errors in the model and in the dataset, and had some insights on how to improve the system. More importantly, although the visualization of trajectories is still an open problem, we found that simple techniques, like the temporal chart, can yield satisfactory results to understand the data. We hope that this tool will help traffic analysts to improve the transportation systems.

9 DEMO AND CODE

- An online demo can be found at http://rodrigonogueira4.github.io/BusData/Outlier_Vis/outlier_v1.html
- A demo video can be found at https://vimeo.com/128097859.
- The source code can be found at https://github.com/rodrigonogueira4/BusData.

APPENDIX

In this section we present old mock-ups that were presented and discussed with Professor Enrico and our mentor Josua Krause throughout this process. Some of them were heavily modified for the prototype in Section 5, but we keep them in this report for provenance.

Mock-up 1: Figure 13 shows a scatterplot that works as an overview for different bus lines. Each dot represents a different bus line, and each axis represents a different query made to the database regarding the outlier detection (total number of outlier points, the distance to the farthest outlier point total

of temporal outliers, total of spatial outliers etc). The user will be able to interact with the axis to change the query they represent. They will also be able to select specific lines and see their corresponding map visualizations. Zoom in/Zoom out is also possible to enable differentiation of points that are too close to one another. This mock-up was slightly modified for Prototype #1 in Section 5.

Why was it discarded? This scatterplot was designed for the concept of our visualization having multiple pages, but was discarded when we transitioned to the concept of having a single page application with multiple views that described different features of a single line.

Mock-up 2: In Figure 14, samples are plotted as a cloud of points. The selected sequence is displayed as diamonds. Each different color represents a different timestamp. This visualization displays only samples that belong to the same bus line. In this example, the user can easily recognize that the selected sequence is a temporal outlier since the selected points are not close to its corresponding cloud of points.

Why was it discarded? This mock-up was abandoned during the Prototype #1, as the visual encoding was not good for our data.

Mock-up 3: In Figure 15, the bus line points are plotted in the map view. It has a timeline which can be marked to show where the buses were at that time, and also, key frames can be marked and by hitting the Play button an animation is shown of the bus movement throughout time, which can also be accomplished by dragging the mouse over the timeline (a more controllable representation of the bus movement). New lines can also be added with the Add Line button. This helps to evaluate different bus lines comparatively, so the user can analyze which of them have more temporal and/or spatial outliers. We ended up not implementing this mock-up because it was too crowded. Instead, we now have an overview for the bus lines (an evolution of Mock-up 1) and different ways of

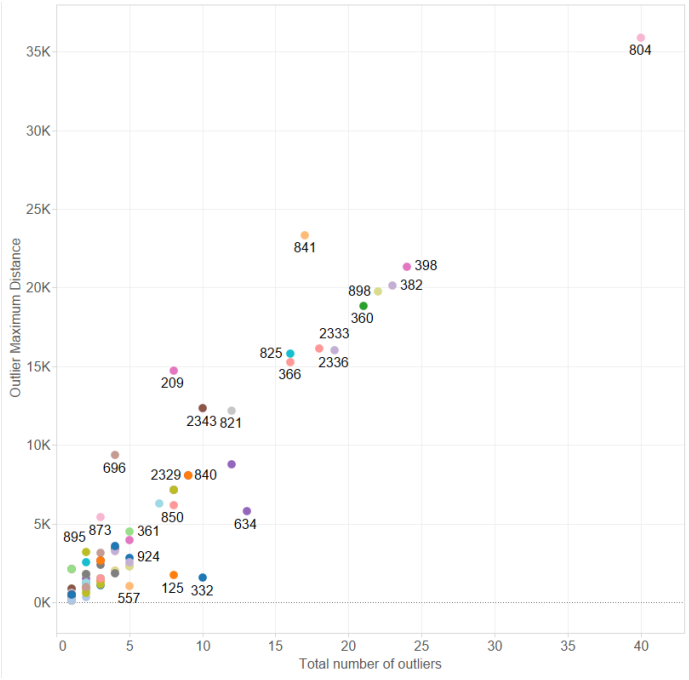

Fig. 13. Mock-up for the overview visualization of our project.

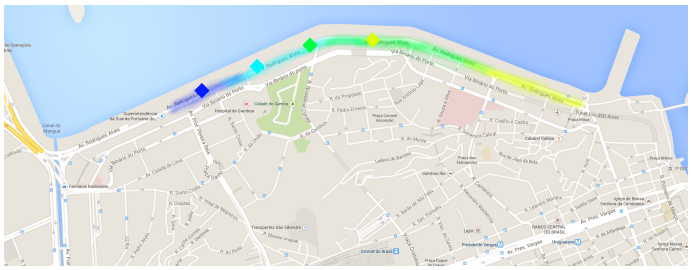

Fig. 14. Initial mock-up for visualization of temporal outliers.

inspecting outliers by using charts and map views.

Why was it discarded? As the project developed, we opted to display the buses progression through time by comparing full trips from final point A to final point B as we felt this was an efficient alternative to explore how their trips vary velocity through time, to create a representation that would better evidentiate temporal outliers.

Mock-up 4: In Figure 16 we explore yet another way of visualizing temporal outliers. Time is represented on the x-axis and the distance from a reference point is represented on the y-axis. In this visualization we can see buses that move slower (or stopped) and buses that are moving faster than the others. This was the clearest way we found to inspect temporal outliers, and was suggested to us by Josua Krause. We ended up integrating it with a general map view for regular buses and outliers, which can be seen in Section 5.

Why was it discarded? This sketch was **not discarded** and was used as a base sketch for the current temporal representation graph.

Mock-up 5: In this Prototype sample, we show the implementation of the scatter plot overview presented in Mock-up

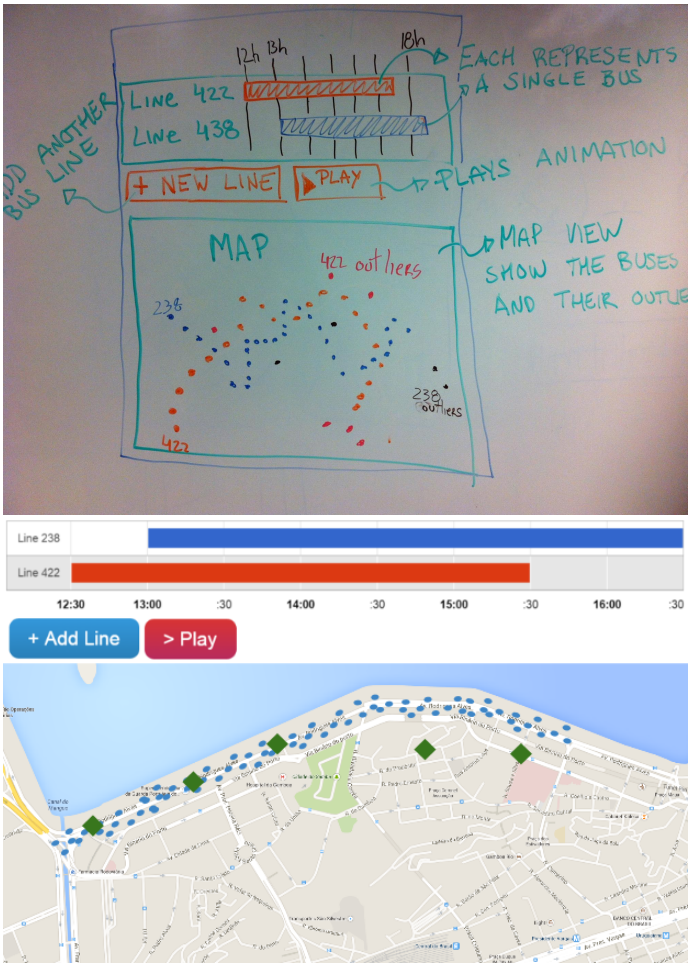

Fig. 15. An example of how the timeline would work with an overview for buses from different lines.

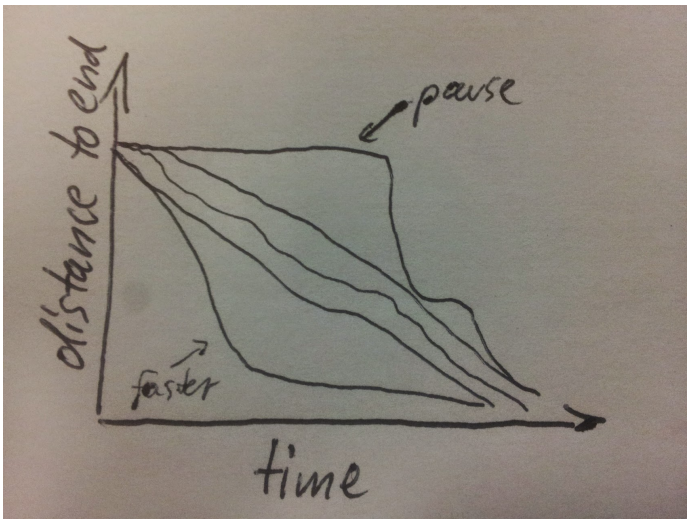

Fig. 16. A cleaner visualization for buses evolving in time and space.

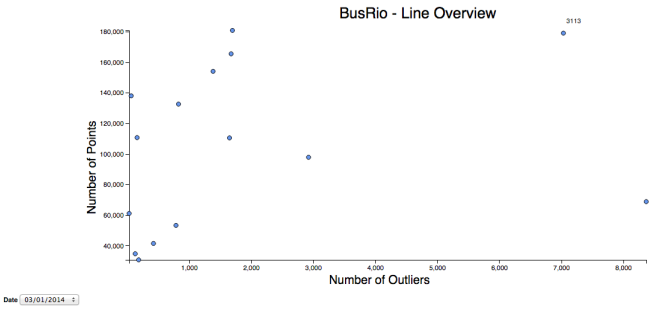

Fig. 17. The overview for bus lines in our system.

1. It encodes, for a single day, each bus line as a point in the scatter plot, with the horizontal axis representing the total number of outlier points, either spatial or temporal, for a line and the vertical axis being the total number of sample points for that line, including the outliers. The user had the options of filtering the date through the combobox in the lower left side.

Why was it discarded? As we mentioned above in Mock-up 1, when we changed our visualization to be a single page design we discarded the scatter plot in favor of other representations, like the two bar charts that already contains the outliers per day and per line.

As one can notice, our sketches and prototypes were very attached to map visualizations. This seemed very intuitive to us, especially because the end-users know the geography of Rio de Janeiro to some extent and may benefit from it when analyzing the outlier information.

References

[1] Le Cun, B. Boser, John S. Denker, D. Henderson, Richard E. Howard, W. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a back-propagation network, In Advances in neural information processing systems. 1990.

[2] Wegenkittl, Rainer, Helwig Loffelmann, and Eduard Groller. "Visualizing the behaviour of higher dimensional dynamical systems." In Visualization'97., Proceedings, pp. 119-125. IEEE, 1997.

[3] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." In Computer VisionECCV 2014, pp. 818-833. Springer International Publishing, 2014.

[4] Barbosa, Luciano, Matthas Kormksson, Marcos R. Vieira, Rafael L. Tavares, and Bianca Zadrozny. "Vistradas: Visual Analytics for Urban Trajectory Data."

[5] Ferreira, Nivan, Jorge Poco, Huy T. Vo, Juliana Freire, and Cludio T. Silva. "Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips." Visualization and Computer Graphics, IEEE Transactions on 19, no. 12 (2013): 2149-2158.

[6] Mitchell, Peter, Colin Ware, and John Kelley. "Investigating flow visualizations using interactive design space hill climbing." In Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on, pp. 355-361. IEEE, 2009.

[7] Tominski, Christian, Heidrun Schumann, Gennady Andrienko, and Natalia Andrienko. "Stacking-based visualization of trajectory attribute data." Visualization and Computer Graphics, IEEE Transactions on 18, no. 12 (2012): 2565-2574.