Visualization of African Conflicts the Past Six Months

Jesper Lund

Abstract— Today only a fraction of the conflicts in Africa are reported by the news around the world. In Janurary 2018, 1551 African conflicts were reported. This report discusses the methodology, implementation and result of a web based visualization tool which enables one to gain insight into the African conflicts the past six months.

Index Terms—Multivariate data, Geo-visualization, Chart visualizations, Clustering, DBSCAN.

1 Introduction

In this report, a conflict is defined as a disagreement between two or more parties where violence is present. Today there are many conflicts taking place around the world and one of the continents highly affected is Africa. The media report major conflicts, but the minor conflicts are often neglected. Also, in the articles submitted by the media, it is difficult to get an overview of how the conflicts are distributed and which areas are most affected. This report discusses the development of a visualization tool, which enables the user to get an overview and gain insight into African conflicts the past six months.

2 Background and Related Work

There are existing visualization tools which address a similar problem. One of them is *Visualizing the Crisis in Syria* [5]. This visualization tool concentrates on Syrian conflicts and lacks an overview of for example how the Syrian conflicts are related to the rest of the conflicts in the Middle East. The visualization tool this report addresses enables the user to get an overview of African conflicts, but also offer the option to select a specific African country and explore information about conflicts in that country.

3 Data

The data used in the visualization is fetched from the Armed Conflict Location and Event Data Project (ACLED) and consists of 8000 objects, where each object has eleven dimensions [1]. The data constitues the conflicts in Africa the past six months and one object corresponds to one conflict. The data dimensions are descriped in Table 1.

Table 1. Description of the data dimensions

Dimension	Description
Id	A unique conflict number
Latitude/Longitude	Geospatial coordinates of the conflict location
Location	The location where the conflict occured (Country/Region/City)
Actor	The primary actor involved in the conflict
Date	The date when the conflict occured
Event type	The conflict type (9 possible)
Fatalities	Number of fatalities due to the conflict
Description	A short description of the conflict

• *Jesper Lund is with Linköping University, Sweden, e-mail: jeslu246@student.liu.se*

4 Method

The visualization tool was developed in several steps which are described below.

4.1 Layout

The first step consisted of creating the layout. The application should contain a non-scrollable and full window sized layout with rows and columns divided by borders. The first row in the application view is the header and consists of a title, a button for information about how to use the application and a search bar, which can be used to search for an African country.

The second row in the application view consists of three columns. The left column consists of buttons and descriptive text areas which are used by the user to filter and cluster the conflicts. In the center column, the geo-visualization is rendered and the right column reserves space for a bar chart, a line chart and a text area which displays information about a selected conflict.

4.2 Data Preparation

When the layout had been created, the raw data was imported from a CSV file, i.e. a file which contains comma separated string values, into the application. After importing, the raw data was prepared for visualization. The values of the data dimensions latitude, longitude, id and fatalities were converted from strings to numbers to be able to perform mathematical calculations. Also, the dates were parsed into JavaScript date objects for easy comparisons and the actor values were parsed into groups to enable filtering by actor group in the application.

4.3 Mapping Coordinates

To be able to draw each conflict as a point in the geo-visualization, i.e. on top of an African map, the data values corresponding to latitude and longitude were projected onto the canvas where the geo-visualization is rendered. This was done by using an external *TopoJSON* file, which also contains information to draw the African country geometries [6]. The use of a TopoJSON file reduced the number of lines of code drastically in contrast to manually implementing the country polygons.

In the same process as the projection, the data objects were mapped to each corresponding point in the geo-visualization to be able to extract information from each point.

4.4 Visualizations

When the data values had been mapped to the canvas, the geo-visualization was rendered by appending the points and the country polygons to a canvas.

Next was to create the charts which constitute the right column of the application view. When the user selects a country in the geo-visualization or search for a country in the search bar, the data corresponding to that country is fetched. This is possible because each point in the geo-visualization contains its data values as attributes.

Two types of charts were created. The first one is a bar chart which shows the distribution of event types. The x-axis contains different event types and the y-axis contains the quantity of each event type.

The second one is a line chart which shows how the number of conflicts have developed the past six months. The x-axis consists of the months September to February and the y-axis consists of the number of conflicts each month.

These types of charts were chosen because they show its data in an intuitive way and also because many are familiar with these chart types. Other chart types, such as a pie chart, were neglected because axis labels did not fit the limited canvas area which led to a less intuitive visualization.

4.5 Filter and Cluster

When all the visualizations had been implemented, functionality for filtering and clustering was implemented. The development of the filtering functionality consisted of the same logic as when the charts were implemented. Due to that all the points in the geo-visualization contains its corresponding data values as attributes, it was possible to fetch all points where, for example, the attribute `country` equals `Egypt`, and store these in an array. When the points had been fetched, all the points except the points in the array were hidden, which resulted in that the points were filtered.

The implementation of the clustering functionality consisted of extracting the latitude and longitude from each data object and constructing a two dimensional subspace containing these values. When the points in the subspace had been assigned to clusters, the quantity of points in each cluster were calculated. Lastly, a color scale was created and the color that was assigned to each cluster depended on the quantity of points in the cluster. Points that were not assigned to any cluster, so called outliers, did not get assigned any color. The clustering algorithm is described more thoroughly in 5.4.

5 Implementation

The visualization tool was developed in JavaScript as an interactive web application, because of the easy access the application online. Also, today there are many powerful JavaScript tools for developing visualizations in the web browser. In this section it is described which techniques and frameworks that were utilized, and what software and hardware that was used during the development. Also, advantages and disadvantages with the used techniques are mentioned.

The application was developed in the integrated development environment Visual Studio Code and mainly run in the web browser Safari. The hardware used when testing and running the application was a MacBook Pro with a 2,6 GHz Intel Core i5 processor, a 8 GB 1600 MHz DDR3 RAM memory and an Intel Iris 1536 MB graphics card.

5.1 Layout and Data Management

The layout was implemented using *Bootstrap*, which enabled a responsive layout [3]. The data, as well as the rendering of the geo-visualization, was managed with the JavaScript library *D3* which uses HTML, SVG and CSS to render visualizations in a web browser [4] [9] [8]. D3 was chosen because of its robustness and flexibility.

5.2 Visualizations

As described in 5.1, the geo-visualization was implemented using D3. The geo-visualization was rendered by creating HTML `path` elements, assigning the corresponding country and point data to each point and appending these to a SVG canvas. The bar chart and line chart were implemented using the JavaScript framework *Chart.js*, which was chosen because of its fast computations, simplicity and pleasing design [7]. The structure of the charts did not needed to be modified and therefor the design was prioritized. The two charts were implemented by creating two chart objects and defining the chart type, the x and y-axis data and the design options. Lastly, the chart objects were assigned to one HTML canvas each.

5.3 Interaction

As described in 5.2, each country and point in the geo-visualization was defined by a separate HTML `path` element, which made interactions possible. The interactivity was implemented by assigning a function for each country and point, which is called when the user clicks on its path element. Also, when the user clicks on a country and this function is reached, the charts are created and drawn using the data corresponding to this country. The charts have less interaction options. It is only possible to hover over a data point to see its value.

5.4 Cluster

The clustering is performed with *DBSCAN*, which is a density based clustering algorithm that can discover clusters of arbitrary shape and handle outliers efficiently [10]. This algorithm was chosen because the clustering objective was to highlight conflict dense areas. The algorithm was implemented with the *jDBSCAN* framework, which enabled clustering by defining the scanning kernel radius, the minimum number of points which constitutes a cluster and the distance method [2]. With a trial and error method, these parameters were set to 2.2 units of length, 80 and Euclidean respectively.

6 Results

The result is an interactive visualization tool which shows African conflicts the past six months. The user is given a number of tools to gain insight into where, how and by who the conflicts occur.

The user can explore details about conflicts corresponding to a specific country by selecting it. This is done by clicking on or searching for the country in the search bar. See Fig. 1. The selecting computations are executed in roughly 0.5 seconds.

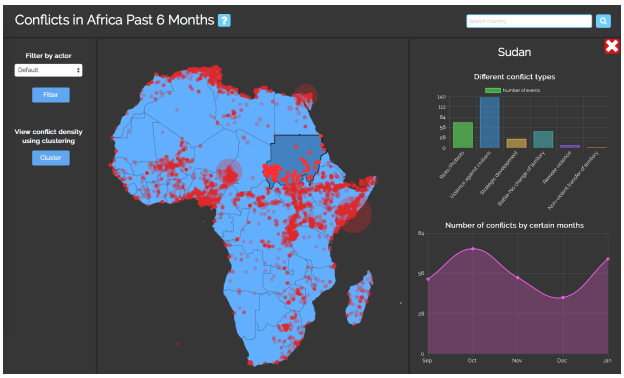

Fig. 1. A state in the application where the country Sudan is selected.

It is also possible to filter the data by a certain actor. In the dropdown list in the left column, the user can select an actor and press the button labeled `Filter` to filter the data by that actor. When filtered, the charts contain data of all African countries with only the selected actor, see Fig. 2. This is executed in about 0.8 seconds.

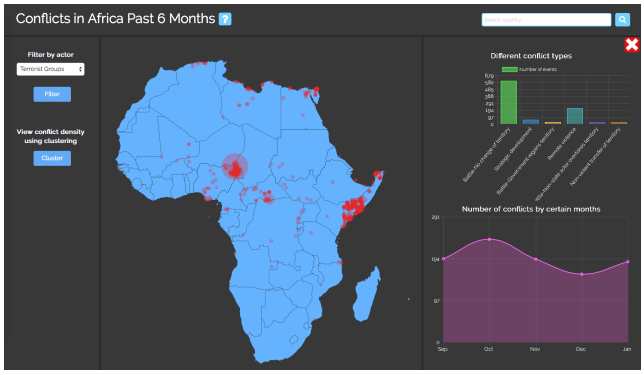

Fig. 2. The conflicts are filtered by the actor 'Terrorist Groups'.

Below the drop-down list is a button labeled `Cluster`, which runs the clustering algorithm. The result when pressing this button is shown in Fig. 3, where one can distinguish conflict dense areas as well as the scale of conflict quantity in the clusters. The clusters with minimum number of conflicts have a light green color and the clusters with maximum number of conflicts have a dark red color. This action is executed in 2 seconds.

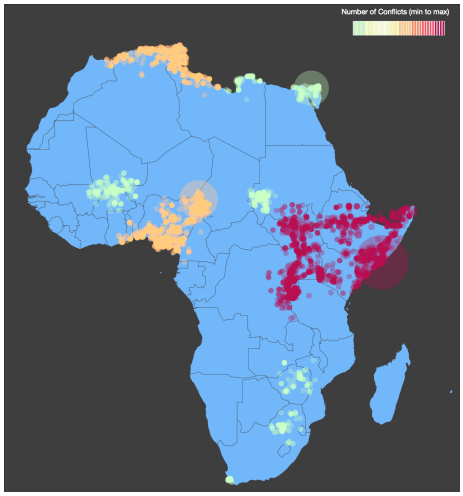

Fig. 3. The conflicts are clustered, which shows conflict dense areas and the scale of conflict quantity in the clusters.

One limitation in the application is the amount of data. Due to that the application is run in a web browser, it is the web browser that manages the data. Currently the data is imported into the application in 2 to 3 seconds. An example of the limitation is when the user moves the map in the geo-visualization, the points' attributes such as position and size, are recalculated and the points are redrawn. Also, the speed of the actions described above is dependent on the amount of data. A larger dataset yields more calculations and thus the importing and computation time increases.

7 Evaluation

When all the application's functionality had been implemented, a user-centred evaluation study with four participants was carried out. The purpose of the study was to evaluate the efficiency and intuitiveness of the usage of the application. The participants, which were technical students at age 20, 22, 25 and 26, were each given four tasks which objectives was to find a specific answer by using the application's different functionalities. The tasks were:

- Which conflict type has occurred the most in Sudan and how many conflicts of this type has occurred?
- How has the number of conflicts in Morocco changed over the past 6 months? (positive or negative)
- In which country has the most conflicts occurred where the actor is United Nations (UN)?
- Which region in Africa is the most conflict dense? (e.g. south, north-west)

The intuitiveness was measured by urging the participants to think aloud, i.e. verbally report what they are doing, believing and thinking while using the application. This resulted in that several unintuitive matters were found, such as:

- Unclear what the different point sizes in the geo-visualization mean
- The points in the geo-visualization is stacked on top of each other, which makes it difficult to interact with
- One can not select a country and then filter the conflicts in the selected country

The usage efficiency was measured by counting how many clicks in the application that were needed to derive the answer. The minimum needed clicks to derive an answer for each tasks was one, one, two and one respectively. The result of this method was that the third task needed about a twofold amount of clicks than needed, due to that the current filter was canceled when the user clicked on another country. This was perceived as unintuitive. The other tasks did not need additional clicks, which insinuates that the functionalities used during these tasks were easy to grasp.

8 Conclusions and Future Work

The result of the application turned out as it was intended. The geo-visualization offers an overview of the conflicts and shows the conflict distribution. Also, the user has the option to further investigate the conflicts of certain countries and actors.

An issue that could have been addressed differently is the data import. Due to that this is the main limitation of the application, a more efficient method than the D3 method currently used could have been used. This could result in that a larger dataset could be used, which could result in a greater insight into how the African conflicts have developed over time.

The D3 library is robust, but also computational heavy. Another library could have been used for the geo-visualization, e.g. *Chart.js*. This would result in a less modifiable map that needs additional lines of code to be implemented, but could yield less interaction delays.

The conflict dense areas can approximately be perceived without clustering the conflicts. Therefor, the conflicts could have been clustered using different parameters than the latitude and longitude which could result in more interesting findings. However, the data set mostly contained categorical values and not numerical values which should be used when clustering with *DBSCAN*. Future work could involve replacing the current clustering algorithm with another algorithm that can cluster categorical data values.

There are a lot of functionalities that can be added, and existing functionalities that can be further developed. One important functionality that was mentioned in the evaluation study is the option to filter conflicts in a specific country. This would result in that the user has the option to gain further insight into conflicts in a specific country. Currently when the user selects a country, the only option is to interact with the bar chart and line chart.

References

[1] Armed Conflict Location and Event Data Project. ACLED - Bringing clarity to crisis. `acleddata.com`, 2018. [Online; accessed 2018-03-03].

[2] Corneliu S. jDBSCAN. `github.com/upphiminn/jDBSCAN`, 2013. [Online; accessed 2018-03-04].

[3] B. Frain. *Bootstrap: Responsive Web Development*. Packt, Birmingham, England, 2012.

[4] M Bostock, V Ogievetsky, J Heer. D3 data-driven documents. In *IEEE Transactions on Visualization and Computer Graphics*, pages 2301–2309, 2011.

[5] Mar Carpanelli, Lily Li, Maria Schwarz. VISUALIZING THE CRISIS IN SYRIA. `syria-visualized.com`, 2016. [Online; accessed 2018-03-02].

[6] Patrick Butler, University of Nebraska at Omaha. Mapping Temporal Datasets with D3. `cartographicperspectives.org/carto/index.php/journal/article/view/1332/1437`, 2015. [Online; accessed 2018-03-02].

[7] S. Powers. *JavaScript Cookbook: Programming the Web*. O'Reilly Media Inc., Sebastopol, CA, USA, 2015.

[8] A. Quint. Scalable vector graphics. *IEEE MultiMedia*, 10(3):99–102, 2003.

[9] J. Spurlock. *Responsive Web design with HTML5 and CSS3*. O'Reilly Media Inc., Sebastopol, CA, USA, 2013.

[10] Z. L. Xudong Luo, Jeffrey Xu Yu. *Advanced data mining and applications*. Springer, Guilin, China, 2014.