Visualization project learning diary 4 - Visualizing Reuters Corpus

Heli Huhtilainen - 3.5.2021

The goal of the week

Last week I created a view of how Reuters corpus document topics were distributed by country. I implemented the view with word clouds and Dash. This week I was planning to deploy my visualization to Heroku. I also wanted to create an overview of the report distributions around the world, as I thought this would be a good initial view in my visualization. I also wanted to make the world cloud view more interactive, if possible.

I also wanted to try how parallel coordinates and bar charts could be used to visualize the data.

Parallel coordinates

First I tried parallel coordinates and parallel categories with Plotly, as I thought they might be an interesting way to visualize the topic amounts in different countries. Some of the test plots are shown in Figure 1. I soon realized that as there were 206 countries and 103 topics, the visualization got too busy with all the data. There would have to be heavy filtering of both countries and topics to get a clear view of anything. Also, the parallel coordinates did not show separate countries in the plot, which kind of was the whole point of the visualization. In parallel categories, the grouping was done according to how many time a topic was used among the countries, which was not useful either. I decided to try something else.

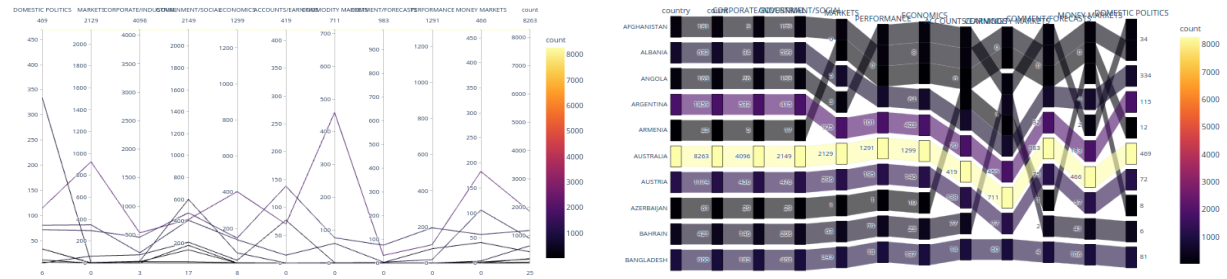

Figure 1: Parallel coordinates and parallel categories both failed to give a good view of how the topics were distributed in each country.

Bar charts

Next, I decided to try bar charts, which work well in showing discrete distributions, like in this case how the document amount is distributed between countries or how the topic distributions differ between countries. They are also effective as the sorted bar chart shows the information both with position and length, which are some of the most effective ways to relay this type of

data. Also, colours can be used to emphasize the effect. First I tried plotting document amounts for countries, and the result that can be viewed on the left side of Figure 2, was ok. I however would have needed to make a separate view to my visualization to show this chart, as the existing view with the word clouds was about the topic distributions by country. I decided to stick to showing the country topics, as the word clouds were not numerically informative on their own, and needed something to augment the information they provided.

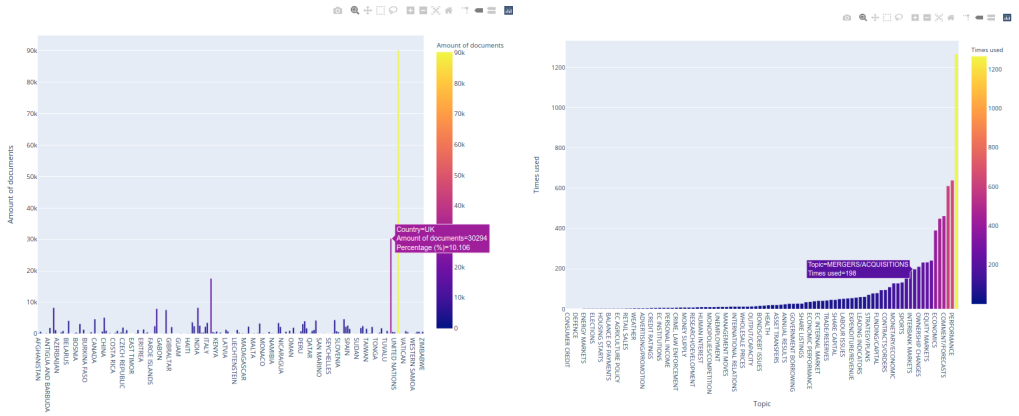

Figure 2: Interactivity made it possible to show document amounts for all countries (left) or topic amounts for one country (right) in one bar chart, even when there were 206 countries and 103 possible topics.

I used Plotly to create the charts and I quite like the result. In the chart view, the user can brush the topics they want to look closer at, and I do not have to implement a separate filtering system. There are also options to show additional details by hovering on interesting details.

Bar chart to the web application

Next, I decided to study how to make the bar chart with Dash. I first managed to make a separate Dash view, and then I proceeded to combine the word cloud and bar chart views. I tried some different options for the layout, and some of them can be found in Figure 3. I decided that it is good to have the bar chart on the left because then the relevant information about the most important topics is situated in the middle. I tried putting the dropbox for choosing the country below the charts, but it was not visible enough there. When situated above the left side of the chart, it does not cover too much of the information even when the dropbox is opened.

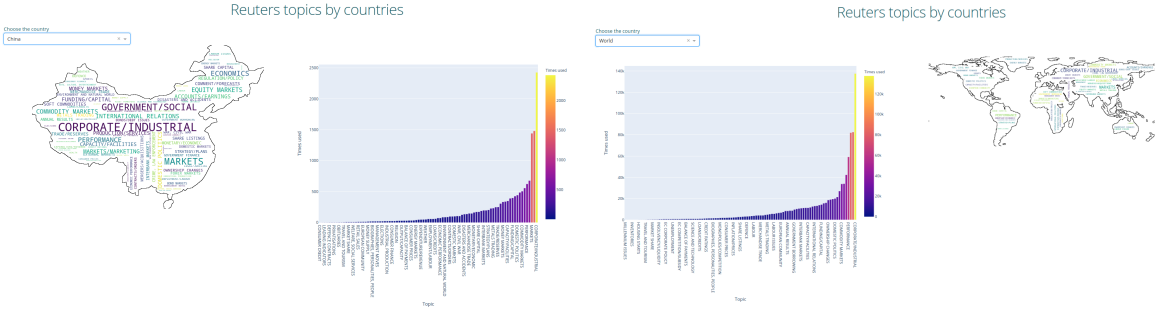

Figure 3: Trying out different layout combinations, the current start view is on the right.

Initially, I had Afghanistan as the start country, as it is the first country alphabetically. However, I wanted to start with an overview of the whole world, so I added a row for the whole world data as the first default view.

After getting my layout more or less together, I deployed my application to Heroku. This part was luckily quite straightforward, as there were good instructions on the Dash documentation (1). I just had to make sure the project structure was compatible and all the relevant dependencies were stated in the requirements. The application can now be found in Heroku (2).

Plan for the last week

Next week I will prioritize writing the project report, but I will do also the improvements that I can manage in the remaining time.

At the moment the colours in the word cloud and the bar chart do not match. It would be good to have the colour scheme of the bar chart reflected in the word clouds so that the bars and the words would be the same colour. The yellow colour might not work in this scenario, so the whole colour scheme might have to be reconsidered. As well as colours, there are also now a lot of different fonts in the view, it would be good to unify them.

I am quite happy with the interactivity of the bar charts. The word clouds however have no interactive properties: the cloud is only a static image. I would like there to be an option to zoom the cloud. It would also be very nice to have the cloud content reflect the changes done by brushing the bar chart. I think these changes might be a bit difficult to implements, so I might not have time to do them.

I would like the users of my visualization to be able to get more information about the visualization if they want. I will add more documentation to the Github page of the project (3), and also add a link to the repository to the application.

Additional ideas

I guess with unlimited time the visualization could be improved quite a lot: there remains a lot of aspects in the dataset that could be visualized interestingly. Eventually, I did not do anything with the actual document text and headlines. Using those would in for example by showing some random example snippets when clicking some topics would make this visualization more interesting.

References

1. Dash deployment: https://dash.plotly.com/deployment
2. Web application in Heroku: https://reuters-visualizer.herokuapp.com/
3. Source code for the project: https://github.com/apndx/ReutersVisualizer