Milestone 2

Devrim Celik, Huan-Cheng Chang, Nina Mainusch

Our visualisation aims at showing how the corona pandemic plays into the recent surge of interest in the Twitch.tv platform. Since our data is so versatile and we have several research questions in mind, we didn't choose to do just one visualisation, but instead we do three. The first one will let the user explore the data as a whole, the second will focus on the pre- and post- lockdown differences and the last one will be about the chess category in particular.

Note that in order to get a glimpse into the implementation process, we chose to write the descriptions and draw the sketches before starting the implementation (as it should be), and then writing the implementation paragraph afterwards, to contrast possible differences.

I. DATA EXPLORATION

We start with letting the user interactively explore our dataset through a bubble chart. For a sketch of the chart cf. Fig. 1. Several information at once will be encoded: every category has its own bubble, the size of the bubble corresponding either to the view minutes, number of viewers/ channels or stream time for a given month and year. The month and year will be adjustable through a slider, so the user can explore himself how the channel loose or gain popularity and how the general trend for the number of channels and time spent on Twitch.tv increases over time. If the user hovers over the bubbles, a little window appears with the name, the logo and the chosen statistic of this channel.
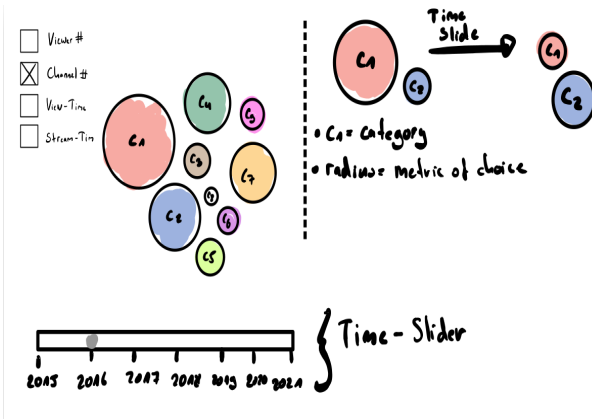

Fig. 1. Sketch of a bubble chart, giving the user a general idea about our dataset.

A. Implementation & Tools

The functions and classes to mention for the implementation of the bubble chart were:

- `d3.pack` and `d3.hierarchy` for transforming our data object into a hierarchical structure and corresponding locations/radii.
- `sliderBottom` from `d3-simple-slider` for the slider.

As it can bee seen in Fig. 5, our bubble chart implementation is almost finished. Bubbles are filled with the image of the corresponding category and if clicked, the bubble in question is highlighted an a `div` appears with a summary of all related data. We introduced a slider for selecting the month and the year, as well as `selections` for language (which we did not originally plan to do) and measure in question (e.g. number of viewed minutes, number of channels, ...).

In terms of further improvements, the only things left are more conceptual questions, such as, whether we want the user to be able to change the number of bubbles, as well as the styling.

B. Lectures

Lectures that come in handy for this plot:

- Lecture 04: D3
- Lecture 05: Interactions, Views (slide 08 onwards)
- Lecture 05: Interactive D3
- Lecture 07: Do's and Don'ts

C. Optional extension

The first plot introduced the user to our dataset. Additionally, if time allows it, we would like to answer the question what the secret behind a successful channel is. Does more stream time imply more viewminutes? Or can you only be successful, if your channel is in a popular category? In order to let the user answer this question, we will create a scatter plot (cf. Fig. 2), where each point represents a category. Through colors, we encode whether a category is in the n top categories or not, where n can be chosen by the user. Furthermore there is a slider for the time as in the first plot, which can also be adjusted by the user.

II. BEFORE-AFTER LOCKDOWNS

Our second plot (cf. Fig. 3) conveys the information of how the corona pandemic influenced the user behaviour of the Twitch.tv platform. Since we are interested in the direct comparison of the pre- and post-lockdown period, we will create an interactive plot with two bars, where the user can choose through a slider which two time points she wants to compare, through boxes which statistics she wants to display and through a drop-down button for which language. If the user hovers over the bars, the exact number of the chosen statistic will be displayed.
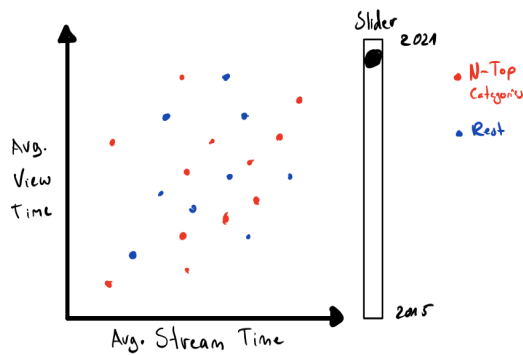
Fig. 2. Sketch of a scatter plot, answering the question what it takes to have a successful channel.
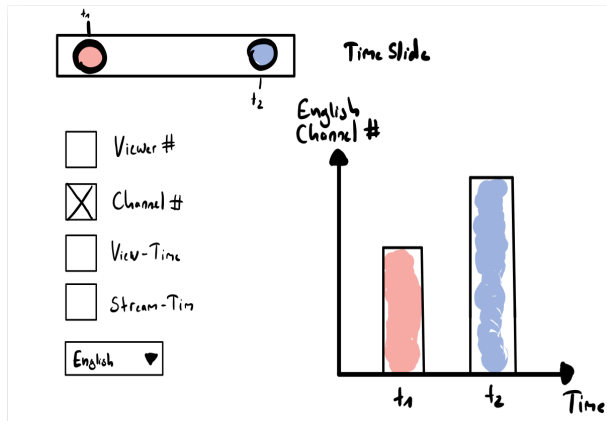

Fig. 3. Sketch of a bar plot, letting the user discover the effect the corona pandemic had on Twitch.tv.

A. Implementation & Tools

There are four parts in this section: the main plot that displays data and three components used to control which data should be displayed. The plot is implemented with `d3.js`, our current implementation can be seen in Fig. 6. It is a basic bar chart with two columns only. The slider is also implemented with package `d3-simple-slider`. Other components are implemented as normal web components in JavaScript. When these elements are changed, the desired copy of data is loaded into the environment and passed to the plot. At this moment, the d3 canvas is cleared and the updated results are displayed.

B. Lecture

Lectures that come in handy for this plot:
- Lecture 04: D3
- Lecture 05: Interactions, Views (slide 08 onwards)
- Lecture 05: Interactive D3
- Lecture 07: Do's and Don'ts

III. CHESS

Our last plot explores the chess category on Twitch.tv. The base plot is a line plot (cf. Fig. 4 left), displaying the trend of

chess: on the x-axis there is the time, on the y-axis the number of viewminutes. Certain events will be marked by boxes, where when the user hover over them, the exact event and some information on it will be displayed.

A. Optional Add-ons

For this plot (cf. Fig. 4 right and left) we thought about an visual embellishment to the plot. It does not contribute much in terms of information, but it will enhance the memorability of our plot. We will add a chessboard and a predefined sequence of moves of three figures (knight, king, queen), where each move corresponds to one of the three events happening. At the beginning the user sees an empty graph, where slowly the timeseries progresses and because the graphs will be linked, each time one of the indicated events happens, the figures on the chessboard move.
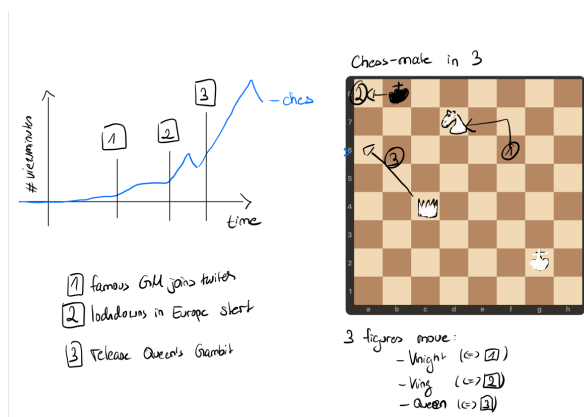

Fig. 4. XXX

B. Implementation & Tools

Regarding the implementation, we use `d3.js` to implement a line chart. Additionally, we add `SVG` components for the three boxes and numbers and on hovering over the boxes and time series, the events and specific values are displayed in detail. The current implementation can be seen in Fig. 7. In terms of what is left to do for the basic plot, we would like to add more visual information to the current info box, which is displayed on hovering over it. Ideally, the time series would also not be static, but start in the left lower corner and slowly develop, where the event boxes only appear when the time series reaches that date. This would make it easier for the user to explore the trend and the underlying events.

C. Lecture

Lectures that come in handy for this plot:
- Lecture 04: D3
- Lecture 05: Interactions, Views (especially slide 24)
- Lecture 05: Interactive D3
- Lecture 07: Do's and Don'ts (especially slide 16)
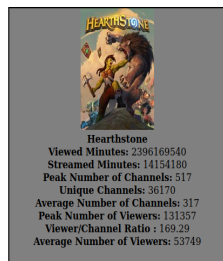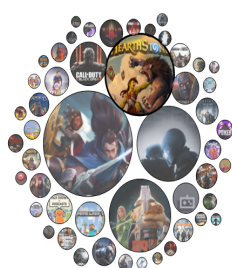
Bubble Chart

All ▾   Viewed Minutes ▾

**January
2016**

2016 2017 2018 2019 2020 2021

Hearthstone
Viewed Minutes: 2396169540
Streamed Minutes: 14154180
Peak Number of Channels: 517
Unique Channels: 36170
Average Number of Channels: 317
Peak Number of Viewers: 131357
Viewer/Channel Ratio : 169.29
Average Number of Viewers: 53749

Fig. 5. Current implementation of our bubble chart.

Bar chart for comparison

English ▾

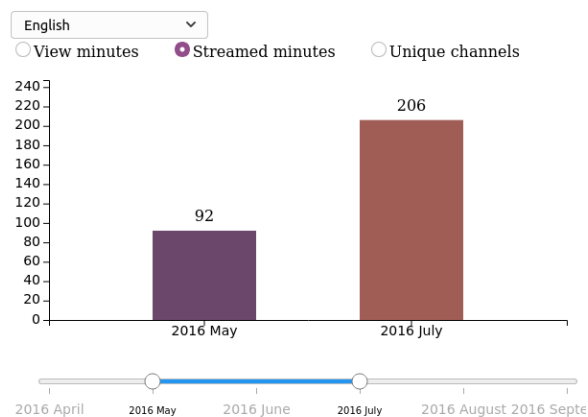○ View minutes   ● Streamed minutes   ○ Unique channels

Fig. 6. Current implementation of our bar chart. Users can select data to be displayed with the sliding bar and buttons.

Chess Timeseries plot

4 July 2017:
Grandmaster Hikaru
Nakamura, 5-times US
chess champion joins
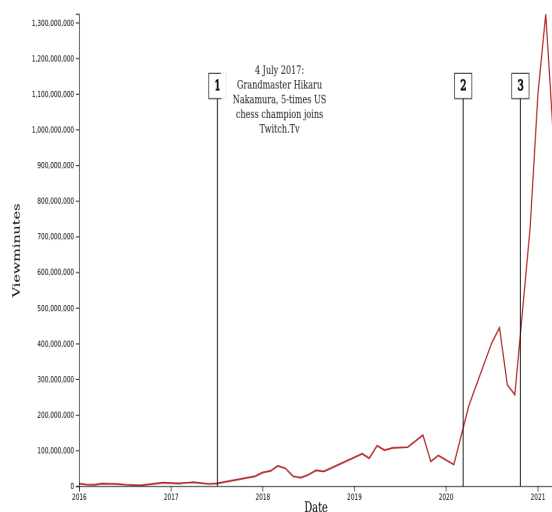Twitch.Tv

Viewminutes

Date

Fig. 7. Current implementation of our line chart. On hovering over the boxes, the corresponding event is described in detail and on hovering over the line, the date and x/y values are displayed.