Differing emoji utilization across communities final report

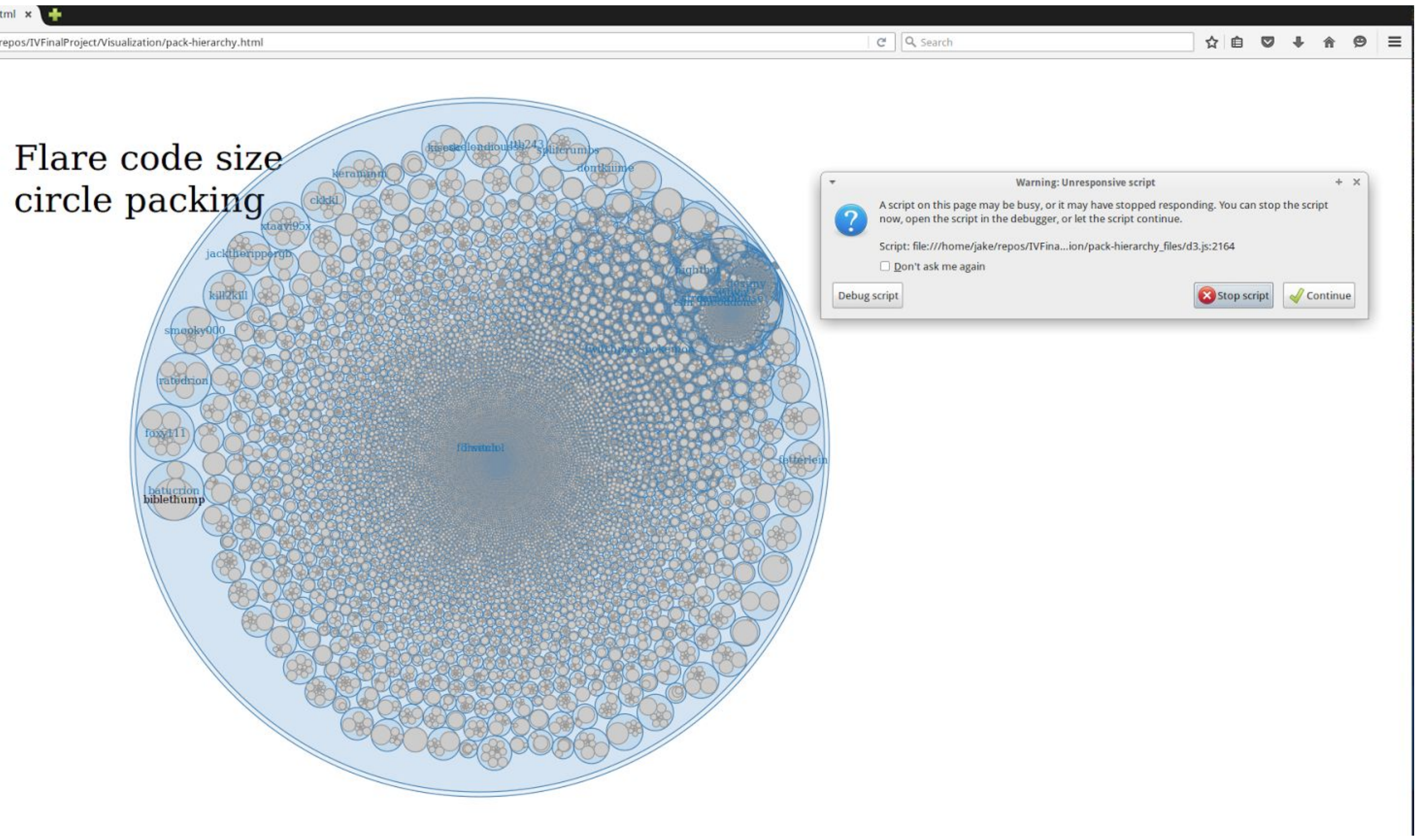
Our central research question was looking at how emoji use changes over time, within different communities or as part of various short trends and memes. We hypothesized that there are a “core” group of emoji that have consistent use over time and the remainder being relatively unused or having peaks of popularity. We also hypothesize that the non-core emoji will be mostly limited to a small number of groups. This would be a useful tool for researchers looking into how ideas enter and spread through social networks, or just for curious individuals who use a lot of emojis.

Our initial plan was to show the emoji usage across groups in a basic bubble enclosure diagram. Those emojis would be surrounded by circles which represent accounts across the different platforms. Those circles could then be categorized inside of bigger circles to represent communities, or trends. The size of the circle indicated the frequency of that emoji or totals of emojis inside that circle. We planned to represent temporal data by allowing the user to click on an emoji, which would bring up a line graph in the corner of the visualization showing usage over time. Up to 3 emoji would be displayed on the line graph and each will have a color red, green, or blue. When the user hovered over the line graph with their cursor, the main circle graph would be colored by relative frequency of the emoji in the circle. This would help us prove or disprove our hypotheses by looking at the color of the main visualization over time. We could also look at the individual line graphs to classify emoji as core or non-core. To make sure this visualization was effective, we would ensure that labels for circles are visible and that emoji appear large enough on-screen. Since we wanted to show the whole visualization on one screen, with a huge amount of data points, we also planned to have a click-and-zoom feature.

When it actually became time to implement our data, we encountered a lot of issues. When we tried to implement the temporal data via the line graph, we found some large gaps in our data. This was probably caused by the fact that twitch had to be recorded continuously in a live stream of data, and any power/internet losses meant a lack of data for that, but it meant our temporal representation had jagged jumps for when we lost data. We considered smoothing the data to make it look nice, but we felt that would misrepresent the data even more.

Another issue we had was with the pure size of the data. With thousands of svg elements the browser would always become unresponsive as shown in figure 1. To fix this, we had to reduce our data considerably. We cut down the number of users, and emojis displayed by pruning our data for only the top elements of each category. We were able to fiddle with the our maximum set numbers for this to balance performance and visualization accuracy.

In addition to this, the packing algorithm we were using, tried to efficiently use space by having differing parent circles encompass the same sub circle. When the visualization reached the size that we demanded, that algorithm failed. It displayed circles wrong, as well as their labels. In figure one you can see that some grey circles have other circles inside of them. This should never have happened. While the sharing of sub circles for parents seemed like a good idea, in the end it hurt us. We had to change our visualization to a more basic packing algorithm, and update how the labeling was handled.



repos/IVFinalProject/Visualization/pack-hierarchy.html
Search
Flare code size circle packing

Warning: Unresponsive script

A script on this page may be busy, or it may have stopped responding. You can stop the script now, open the script in the debugger, or let the script continue.

Script: file:///home/jake/repos/IVFina...ion/pack-hierarchy_files/d3.js:2164

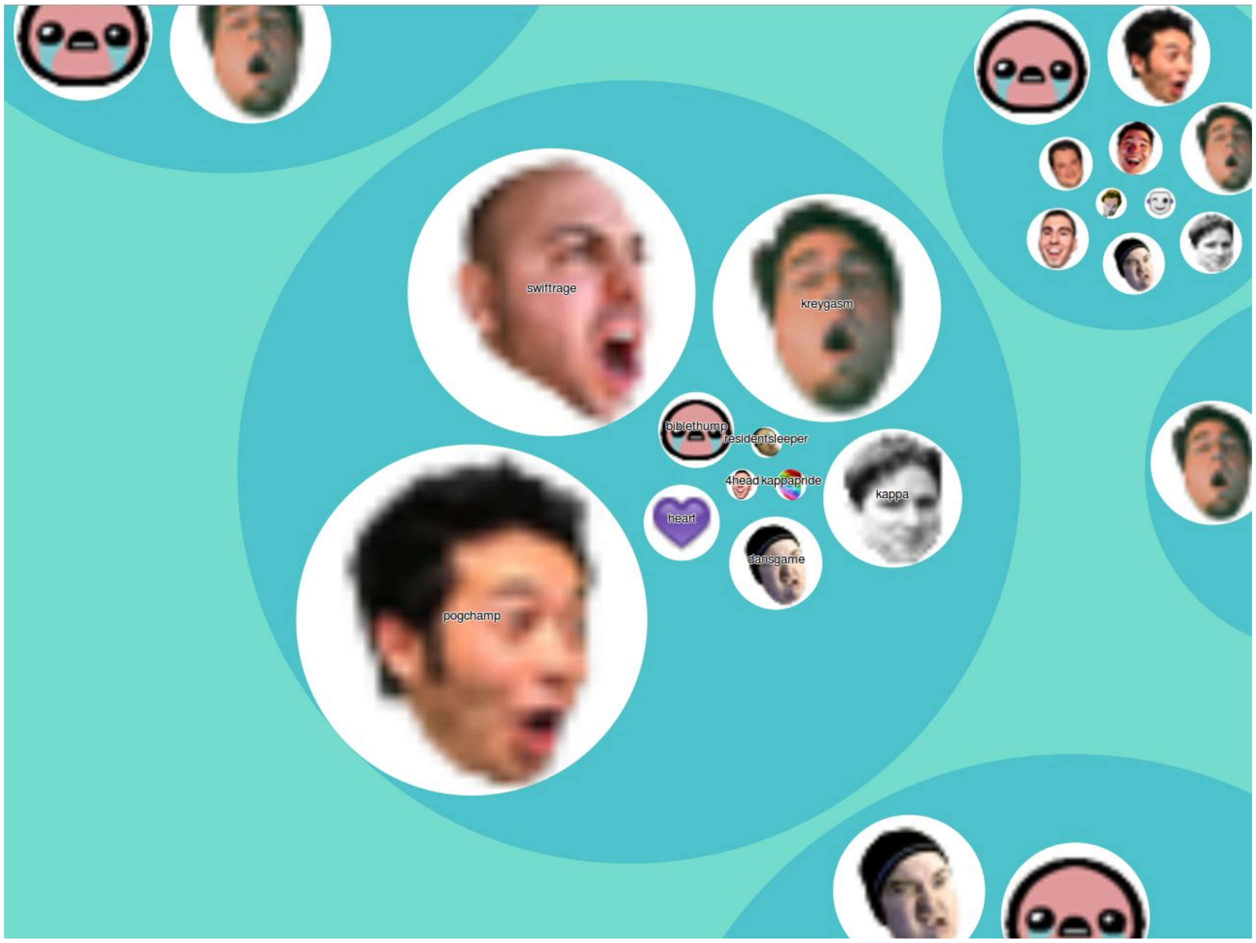
☐ Don't ask me again

Debug script
Stop script
Continue

Figure one: Shows incorrect circle packing and the browser becoming unresponsive.

While using our visualization, we also saw the same emojis coming up time and time again. In this way we didn't see that much difference between the different groups. We considered removing the top used emojis to help users identify the frequent emojis that were unique to a specific group, but this would skew the data and portray an inaccurate representation. To counter this we added a table to the side of the page that showed the top emoji's and their counts. This way analysts could identify and subsequently ignore the emoji's when looking for emoji usage that was specific to individual channels.

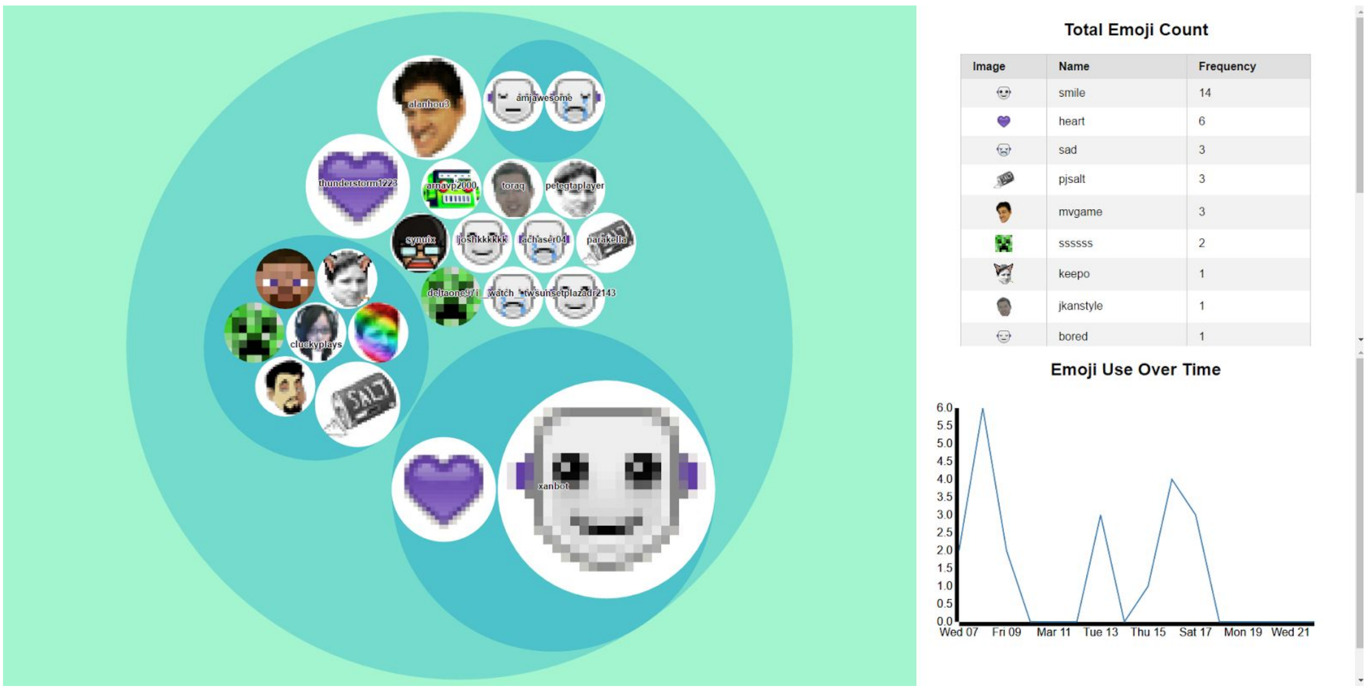
The last major issue we encountered which were unfortunately unable to fix was the low resolution of the images we used. Because the images were only meant to be showed at a low resolution in chats on screen, the low resolution was amplified when users zoom in our visualization. We were unfortunately unable to find higher resolution images for all the emojis we would need to use. This is demonstrated in figure 2 below.



The majority of our classmates that we received feedback from knew what twitch was, and everyone knew facebook. Almost everyone we talked to was interested in the emojis usage of their peers, so we felt that this classified them as the perfect target audience.When we showed our visualization to the rest of the class, we were still fighting the browser crashing issue, but we still received a lot of great feedback.

For one, our color scheme was not easy on the eyes, and did not set apart the user circles from the community circles as they were both the same color. We were also told that we should make the visualization take up more of the screen or just go fullscreen, as the labels were hard to read. This was exasperated by our bad labeling at the time, which we have since fixed. The biggest and most frequent feedback we received was that we should have our bubble packing visualization update live over time instead of using the line diagram. While we agree this is a fantastic idea, we felt that we didn't have the javascript expertise or the time to complete it before this paper. If we were to continue this project, that would definitely be the next step.

We used a few different tools, to gather our data. Jake wrote a quick IRC script to record twitch chat and rob wrote a facebook api scraper. Because of the large size of the data, jake wrote a c++ parser and reducer for that data instead of coding it in python. In the end this was a mistake, because it took quite long to write the parser for the differing outputs of facebook and twitch, but not so long to actually process the data. He could of saved a lot of time had he just used python. Because of the complexity of the Facebook Graph API, rob wrote a small NodeJS script that requested a fixed number of the most recent posts from a public Facebook group from a single API call. From there, messages were sorted by user. Separately, emoji data for Facebook was scraped from fbicons.net. This involved copying HTML and CSS files out and removing a bulk of the text using both alt-dragging for aligned data and regex. All the fields in the HTML file were aligned, but mapping the image classes to the CSS file were unordered and non-sequential. Because of this, 3 separate CSV files were generated for emoji representation and a Python script was written to load and map these files properly. Additionally, the images were not separate, but contained all ~400 images across two spritesheets (an image that contains a grid of other images). ImageMagick and the Python wand library were used to split the spritesheets up into individual images. The text representations of the emoji were also scraped off the website and used to prune messages that did not contain any emoji. For our actual visualization we used the javascript library D3, and Mike Bostock's bubble Hierarchy visualization as a baseline. From there we added our json data, and started updating the visualization to include emojis, better coloring, a better packing algorithm, and totals and temporal data on the side.



Total Emoji Count

Image	Name	Frequency
	smile	14
	heart	6
	sad	3
	pjsalt	3
	mvgame	3
	ssssss	2
	keepo	1
	jkanstyle	1
	bored	1

For splitting up of work. Jake worked on scraping the twitch data as Rob worked on scraping facebook's data. Jake wrote all the parsing and data reduction code, and the twitch-json conversion code. He set up the basic visualization with the data, rewrote how the cicles were packed, and fixed the labeling for large numbers of items. Rob scraped the emojis from a single giant image file that contained all of them, did the final website styling, and implemented the charts, and temporal line graph.

References:

While we specifically are using a enclosure diagram, we feel a paper on general D3 use was the right direction to go as we would be adding new features to common bubble enclosure visualizations. For this reason, we picked the D3 Cookbook as one of our references.

http://www.spatialcapability.com/Library/FOSS4G/Data%20Visualization%20with%20D3.js%20Cookbook.pdf

Zhu, Nick Qui. *Data Visualization with D3.js Cookbook*. BIRMINGHAM - MUMBAI:
PACKT, 2013. Digital.

Another paper we picked was the sentiment of emojis paper. They used multiple visualizations to display the connotations associated with different emojis. It is similar to what we even if they don't use an enclosure diagram in the paper.

http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0144296

Kralj Novak P, Smailović J, Sluban B, Mozetič I (2015) Sentiment of Emojis. PLoS ONE 10(12): e0144296. doi:10.1371/journal.pone.0144296

The final reference that we have is a blog post from the Instagram engineering team showing some of the data they gathered when they were implementing emoji as hashtags. They don't quite use an enclosure diagram but do have a chart of emojis reduced from 100 dimensions to 2 using t-SNE (similar to PCA)

http://instagram-engineering.tumblr.com/post/117889701472/emojineering-part-1-machine-learning-for-emoji

Instagram Engineering. "Emojineering Part 1: Machine Learning for Emoji Trends." *Instagram Engineering*. Instagram, 7 May 2015. Web. 7 Apr. 2016.

Bostock, Mike.*Pack Hierarchy Visualization*.*Mick Bostock*. N.p., n.d. Web. 5 May 2016.
<https://mbostock.github.io/d3/talk/20111116/pack-hierarchy.html>.