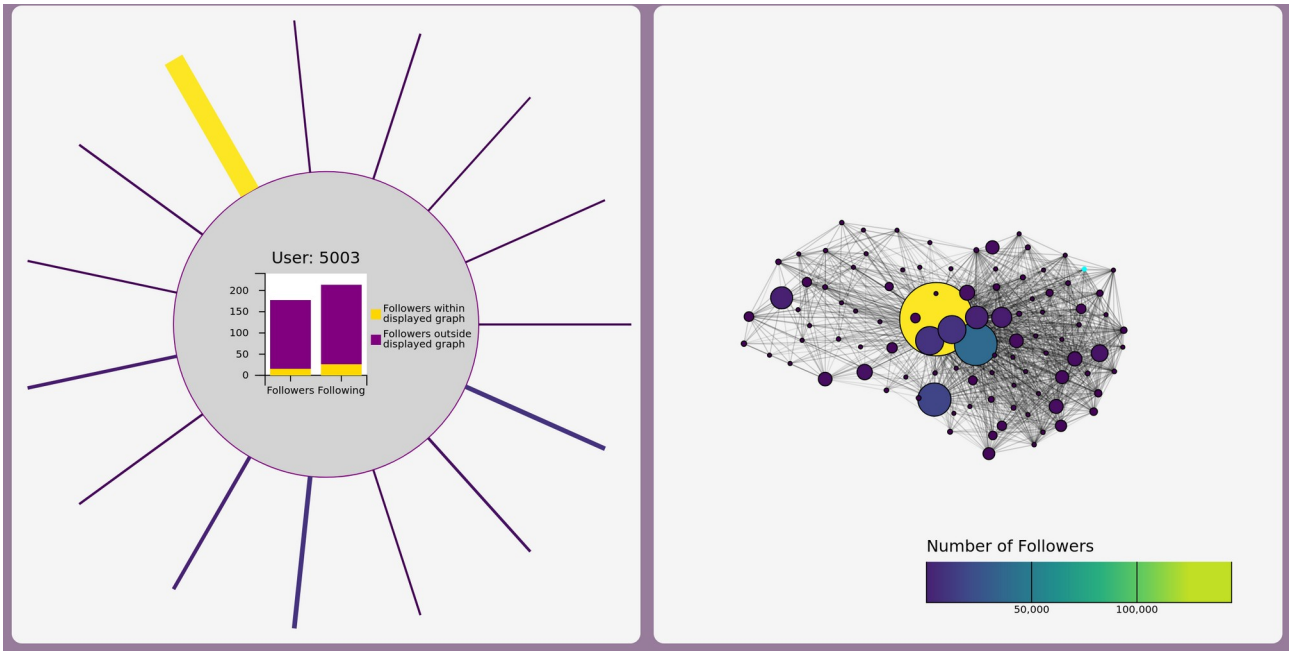
Bluesky Network Data Visualisation

Candidate Number: 1073375

I declare that, except where otherwise indicated, this mini-project is entirely my own work, and that it has not been previously submitted and/or assessed and is not due to be submitted in its entirety or in part for any other course, module or assignment

Bluesky Network Data Visualisation



My project aims to provide a visualisation of the network data from the Bluesky social media platform. Bluesky has a somewhat decentralised design, but is otherwise similar to X/Twitter, and has risen in popularity over the last few months. My project aims to display the data surrounding two subsets of the overall dataset – the most followed users in the platform, and the users with the earliest posts on the platform. It does this by showing an overall representation of the selected subset in the right view, and showing more detailed information on the currently selected node in the left view. This project is intended for a casual audience, who would like to see the connections between the users with the most followers and the users who posted the earliest in the dataset.

The Data

Source: Failla, A., & Rossetti, G. (2024). Bluesky Social Dataset [Data set]. In Plos One. Zenodo. https://doi.org/10.5281/zenodo.14258401

https://zenodo.org/records/14258401

Domain-specific description: The dataset as a whole consists of all data related to the users in the dataset from before March 21st 2024, and all data related to the various custom feeds available in Bluesky at that time. The data used in my project initially consisted of a list of every follow between two users on the platform, and a table of every post made on the platform, alongside all the data related to that post like the number of likes, root thread node, date posted, etc. The final dataset used in the project is the total likes for each user, their followers inside the relevant subset, and the number of followers outside that subset.

Abstract description: The dataset used in the project initially consists of a join between a network dataset and a table dataset. The network dataset consists of two data types: items, represented by a unique numerical number; and links, represented by a pair of numbers. The table dataset contains a variety of data types, including the number used as a primary key for each item, a number used as a foreign key to connect to the network dataset, and various attributes about each post.

In the pre-processed data used in the project, the dataset consists of a table where each item represents a user, alongside various attributes related to that user and a list of every user in our derived subset that follows the specified user.

Pre-processing: The pre-processing required for this project was extensive, with the full dataset taking up over 180GB on my computer once it was decompressed. The majority of the pre-processing was done in Python, with a small amount done dynamically in JavaScript while the program is running. The pre-processing consisted of first converting the list of follow relations into a more useful dictionary format, and then deriving a list of the top 500 most followed users. A list of the 500 users with the earliest posts was also derived from the post data, and then their follow relations were extracted from the dictionary. For both subsets of the data, the follow relations were then trimmed to only include the users in that subset, with a count being maintained of all eliminated followers. The total number of likes each user had was also derived from the post data. Later in the project, I also derived a dictionary of every person who is followed by the users in our subsets. All relevant data was then converted to the JSON format and included in the data visualisation. Some pre-processing is done in the JavaScript to dynamically reduce the size of the graph further.

Goals and Tasks

Domain-specific tasks:

1. Compare the follower graph of the most-followed users with the graph of the earliest users to post on the platform, and change the size of the subset being shown.

2. Browse the various users included in the most-followed users or earliest users, and see relevant information about that user like number of followers, and total likes on all their posts.

3. Compare the number of followers a user has to the number of people followed by that user, both inside and outside of the shown graph.

4. See how many followers each user has, alongside which other users are followed by that user, and compare that to other users in the graph.

The attributes being visualised in this project are the following:

- The subset of the data being viewed (both the category of the subset and its size)
- The number of followers that a user has outside of the shown graph
- The number of followers that a user has inside that graph
- The number of people the user follows outside the graph
- The number of people the user follows inside the graph
- The total number of likes a user has gained on all their posts.

Overall, the aim of the visualisation is to allow the user to compare the social graphs of the two subsets by exploring the data surrounding each graph.

Abstract description of tasks:

1. Compare two subsets of a larger dataset, and modify the size of those subsets.

2. Explore the graph of the items in the various subsets of the dataset, and view the attributes related to each node.

3. Compare the number of incoming and outgoing links that each node in the network data has, both within our subset and outside of it.

4. Provide the ability for the user to discover and enjoy information about the different users and how they connect to each other via the network data.

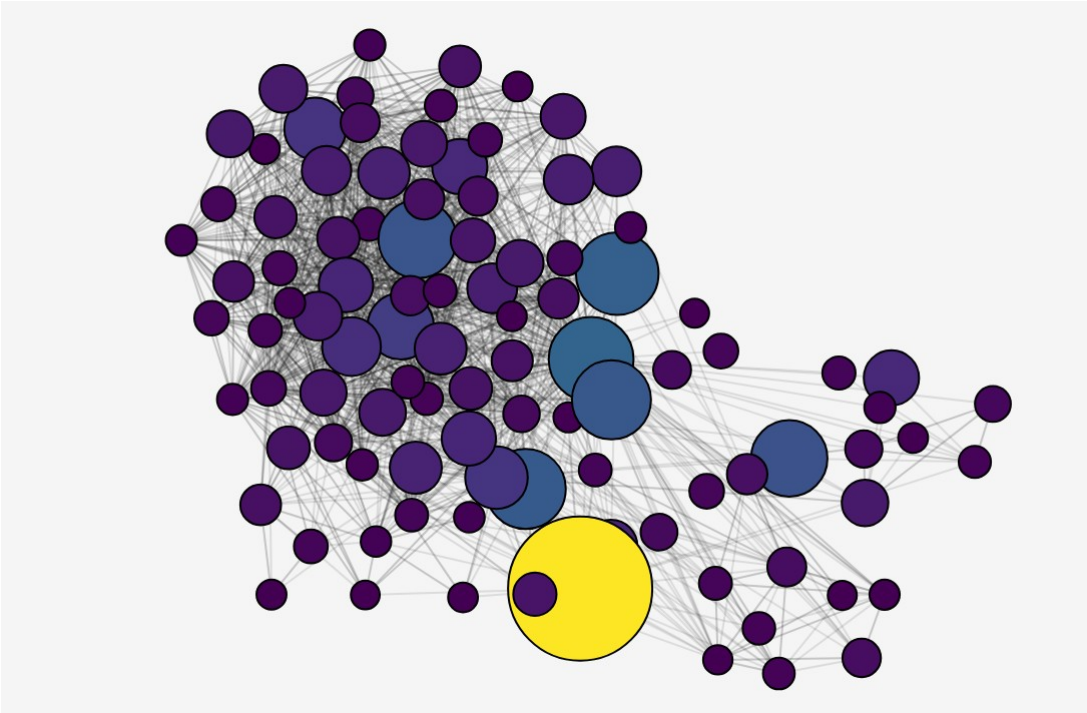
There is also the capability for the visualisation to be used to search for certain items, such as the item with the most incoming links, or to locate a certain item if its primary key (user ID) is already known.

There is also a degree of summarisation done within the graph, and it should be possible to get a sense of the data surrounding followers by looking at the network data without any user interaction.

The main target of the visualisation is to show the topology of the network data, alongside showing various trends within the data.

Visualisation Design

View One: Force Directed Graph



Description:

A graph representation of network data, where the nodes are mapped to circles and the links are mapped to lines between the circles. Each node is given a certain “charge”, and each link is given a certain “elasticity”. The display is then iteratively simulated according to the forces of repulsion and attraction present on each node, resulting in an animated display that eventually reaches an equilibrium. The size and colour of the circles are determined by the data related to each node, with a legend being shown to explain the colour scheme. In my project, the nodes represent a Bluesky user, the links represent one user following the other, and the size and colour of the node both encode the number of followers the user has.

Interactivity:

There are 6 different interactions programmed into the visualisation:

- Drag: The nodes are able to be dragged around the screen, updating the forces on all the other nodes accordingly. This allows the user to change the arrangement of the nodes however they would like.
- Click: Upon being clicked, that node is selected, and becomes the focus of the second view of the visualisation. The node’s colour is also changed to signify it has been selected. This allows the user to select a node to gain more information about it.

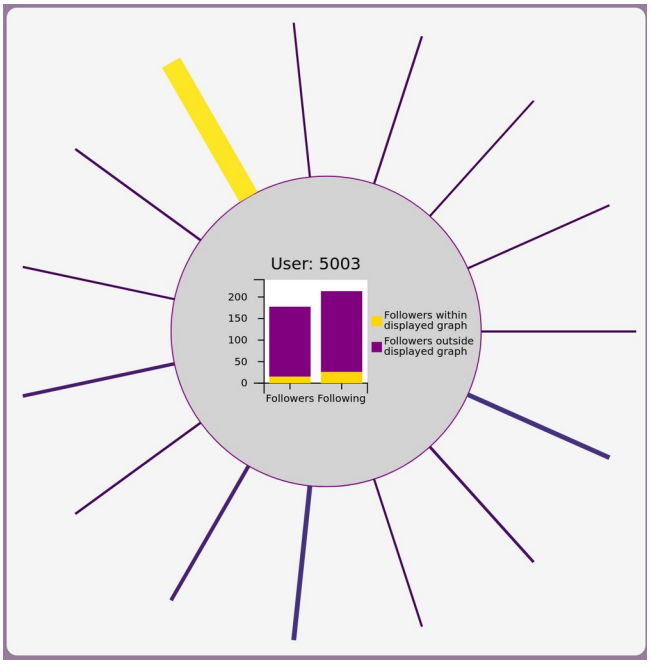
- Hover: When hovering over a node, a different selection type is applied, where the node is given a larger outline of a different colour, and all links to the node are coloured orange. The hover interaction is coordinated across both views, so hovering over a user on either view selects that user as the hover focus for both. A tooltip also shows the user ID.
- Scroll: You can use the scroll wheel to zoom in and out of the graph.
- Slider: This applies a semantic zoom to the dataset, cutting down the number of nodes being simulated to the top n many nodes.
- Drop-down menu: This menu changes which dataset is currently selected. This could be considered either a semantic zoom or a type of constrained navigation.

Rationale:

This view is designed to give the user an overview of the data available in the visualisation. It is difficult to find a stable and visually pleasing representation of the entire graph, so a force-based simulation, one that the user could interact with, was chosen to be the idiom for the graph overview. The nodes are coloured according to the Viridis colour palette, which ensures that the graph remains colour-blind friendly, and the accent colour for the links is orange for the same reason. Size was chosen to encode the number of incoming links a node has, as displaying that is the primary task of this view. However, the radius of the circle is proportional to the square root of the number of incoming links, as the size encoding is one of area, not length. An attempt to make the colour channel encode a user's total number of likes proved only to be confusing, so colour instead redundantly encodes the number of followers a node has.

Click was chosen as the primary selection idiom as hover resulted in too many changes to the other view, which looked jarring. Hovering is generally best suited to browsing tasks, and so is primarily used to help present connections and identifying data. The slider is present as the graph is too dense for a simulation of all 500 nodes in the dataset to be feasible on most machines – a default of 100 was set as that was both a reasonable number of nodes to visualise, and wasn't too taxing on my machine. Furthermore, the semantic zoom/filter this provides helps to reduce the complexity of the visualisation, as the simulation with all 500 nodes is near incomprehensible. The drop-down menu was included to allow the comparison of the two subsets without having to simulate them both simultaneously.

View Two: Novel Visualisation/ Spider Graph



Description and Analysis

This view allows the user to see more detailed information about a node of their choosing. It consists of a circle containing the user ID and a stacked bar chart, and multiple lines, or 'legs', extending from the circle. For the stacked bar chart, the marks are rectangles, and the relevant channels are the x position, height, and colour. Each rectangle encodes information about a certain subset of links connected to the selected node. The x position encodes whether the rectangle represents the number of people the selected user follows, or the number of people following that user. The colour encodes a different category, which is whether or not the links it represents are inside or outside of the displayed graph. The height encodes the number of links in that rectangle's category. Overall, it is a bar chart where height encodes the number of users that the selected user follows or is followed by, and the different stacks represent the number of links inside or outside the graph in that category.

For the 'legs', the marks are the individual lines extending from the circle. The channels of position and tilt are used to differentiate each leg, as each leg represents a different node in the dataset. The channels that each node's data is encoded through are width/thickness and colour. Both channels encode the same data, which is the number of followers that the user has.

Interactivity

- Hover: Two different things are encoded via hover in this view: the display of a tooltip, and the emphasis of a given 'leg'. A tooltip is displayed when the user hovers over the stacked bar chart, and displays all the available information about the selected user, including the total number of likes their posts have, and the number of people they follow or are followed by. A tooltip is also displayed when hovering over one of the legs, showing the user's ID alongside some information on the user represented by that leg. Hovering over one of the legs also increases the

width of that leg, alongside emphasising that node in the other view, making this a type of selection interaction.

- Click: Clicking on a leg is a selection interaction which changes the focus of the visualisation to the user represented by that leg. This selection is also linked to the click selection in the other view.

Rationale

This was initially intended to be a combination of the small-multiples/ glyph-map idiom and a network idiom, but the combination of the two proved to be too cluttered for a single view. A stacked bar chart was chosen to be the central glyph, as it allows both the proportion of in-graph and outside of graph followers to be shown. It also allows for a comparison of the number of people that the user follows and is following. Unfortunately, all the users shown in the dataset tend to follow significantly less people than they are followed by, so the comparison between bars isn't always possible to achieve visually. This is the rationale behind having a tooltip show the exact user data regardless of which bar the user hovers over.

The channel of tilt is used to differentiate different nodes, as it is difficult to encode other information with it, but every mark is still easily differentiable when used in this way. Redundantly encoding line width alongside colour was done because the lines were too thin for the colours to be differentiable from one another, and so thicker lines are more easily distinguished, and thinner lines are less likely to be the priority of any analysis. Either way, the tooltip also allows the user to see the information relevant to that leg. The colour scheme used for the legs is the same one used for the other view, and is a colour-blind friendly palette.

Overall, this view is appropriate for the tasks involving individual users' data, as it acts as a more focused view on a single user. However, it still displays the network data in a way that is clear and relevant to the user.

Visualisation Principles

There are a number of reasons why I believe my novel visualisation is more suited to the tasks present than any of the encodings given in the course materials. It allows the user to see detailed information about a given node, while also displaying all the links from that node in a straightforward manner. It encodes data using a variety of channels, and saves the more discriminable channels for the more important data. None of the network idioms shown in the course allow a large amount of information to be displayed per node (unless you use a tooltip), and so they wouldn't work well with the tasks the novel visualisation is built for.

All of the other encodings provided in the course materials do not allow for the network data to be represented in a meaningful way, so navigating from one node to another would require using a different view. In the case of my visualisation, this would be particularly difficult as the graph is fairly dense, so switching from node to node would prove to be rather fiddly.

A stacked bar chart also appears to be the best encoding for the individual node data. Since there is such an emphasis on the proportion of incoming links that are from inside the graph to ones that are outside the graph, a visualisation that allows the user to view proportions is necessary. A pie chart would accomplish this task, but then there would be no encoding for the total number of incoming links outside of just writing the number. Furthermore, there would be no clear way to encode the number of outgoing links from that node.

A scatter graph could somewhat encode the two categories of incoming links, but then finding the total number of incoming links would be difficult. Furthermore, a scatter graph requires two continuous axes, and one of our axes is discrete. This problem would be even worse with a line graph, as the lines between nodes would violate the expressiveness principle.

A heatmap would only encode the data in terms of colour, which is less discriminable than aligned length.

The other idioms shown in the course materials require more data than an individual node has in our dataset, and so wouldn't work well for our visualisation.

Furthermore, if there were enough data related to an individual node that another idiom would work better than a stacked bar chart, it would be very easy to swap out the bar chart for something more relevant, while still maintaining the network aspect of the novel visualisation.

Overall, my visualisation allows for the user to perceive the network data related to the node alongside key information about the node in a clear and concise manner.

Credits

There were multiple times the course materials were used in my visualisation:

- The layout for the entire implementation was loosely copied from the tutorial answers
- The file loadAndProcessData.js was initially copied from Sheet 4, Question 1, although major changes were made to how this file works.
- The file barchart.js was copied from sheet 5 question 1 and edited heavily to make it work with my data. The implementation of stacked bars was notably not copied from any of the course materials.
- The file colourbar.js was copied from sheet 4 question 1 and had some minor tweaks made.
    - The implementation of the viridis gradient in the colour legend was based on https://observablehq.com/@tmcw/d3-scalesequential-continuous-color-legend-example
- The force directed graph was heavily based on the force diagram available in the course materials, although it was tweaked extensively.
    - The implementation of draggable nodes was copied from https://observablehq.com/@d3/force-directed-graph-component

There were also many times I referenced stack overflow for help with certain bugs or problems I was having. I usually didn't directly copy the code, but the one time I copied an answer extensively was with converting an object to an array, where I used the answer from https://stackoverflow.com/questions/38824349/how-to-convert-an-object-to-an-array-of-key-value-pairs-in-javascript

The code for the slider was copied from https://www.w3schools.com/howto/howto_js_rangeslider.asp and modified to make sense in the context of my program.