Information Visualization
Sean Carey, Carol Chen, Yo-Shang Cheng, Anjana Dasu
http://bit.ly/twitterSuperstars
May 10, 2010

Twitter Superstars: Final Report

Introduction

Twitter has emerged as an extremely popular social networking and microblogging platform. Millions of people—celebrities and non-celebrities alike—use Twitter. Its API (application programming interface) has made it possible to write applications for studying human tweeting behavior. Many visualizations study how Twitter users are related in networks, or study the content of tweets; however, as far as we know, nobody has created an exploratory visualization for visualizing people's behavior and level of activity on Twitter.

We wanted to use the skills we acquired in class to visualize some measure of success on Twitter. Shape is a very important aspect of this. We subconsciously interact with our world, taking into account the shapes of objects that we interact with. For our TwitterSuperstars visualization, we wanted to take advantage of people's familiarity and comfort with shapes and turn something conceptual, like Twitter statistics, into something more tangible. Humans can better describe their impressions of others though visualizations than with statistics. Our premise was that there is some reason behind the success of movie stars, musicians and TV personalities on Twitter, and perhaps a reason why they have millions of followers. For example, we guessed that comedians might have more followers because people may like to read funny tweets. Therefore, we asked ourselves, what does celebrity success on Twitter look like? Might their offline success carry over to Twitter, and might they be able to maintain it by using Twitter in an engaging way?

Initially, we talked about inspecting the content of tweets and using that information to figure out what types of content make for successful Twitter users. We soon realized that such a task would be very natural language processing (NLP) driven and did not work well given the limited time frame and scope of the class. Instead, we discussed other ways of measuring success and developed criteria that might be representative. This included celebrities' occupations, number of followers, number followed, tweets per day, number of links, replies, mentions, retweets, and photos, and use of emoticons, hashtags, and exclamation points for describing excitement.

We chose to use parallel coordinates because we had many variables to visualize and thought it would be useful to filter and see only the line shapes for celebrities that reply often, or for celebrities who use few hashtags. We also did not come across parallel coordinates in any other Twitter visualization. We felt that the parallel coordinate system would be beneficial in presenting the information to users, because it allows individual tracking of information in a way that easily allows exploration and filtering of the data.

Data Set

We decided to visualize the behavior of celebrity Twitter users for a number of reasons. First of all, we felt that people would want to make comparisons between people they know, or at least recognize. A Twitter visualization like the TwInluencers

Visualization[1] is interesting, but it does not encourage comparison, and we hardly recognize anyone in the set of "twinfluencers" that it visualizes. Secondly, celebrities work well because their activity is public. It would be nice to analyze the behavior of people we directly know, but we may run into privacy issues if we use our friends' Twitter feeds. Thirdly, there are a variety of types of celebrities on Twitter, and we thought it might be interesting to compare across groups, like actors and politicians. Finally, we wanted to work in a defined space, but with all the data available it can be difficult to know what to look for. Many people compile lists of celebrities to follow on Twitter, so we leveraged this information.

We started off by using Mashable's celebrity Twitter list (Mashable is a blog focusing on social networking), which is a nice size at 93 users, all of which are verified as officially belonging to celebrities. A few more athletes and politicians were added to this list to round out those categories.

Using a combination of Python scripting and the Twitter API, we gathered the most recent 3000 tweets for each user (a limit of 3200 is placed by the API). Although some of the celebrities had more than 3000 tweets, we figured that 3000 should give a pretty good sample of their typical behavior. In addition to the tweets themselves, the API gave access to information such as the number of followers celebrities have, the number of people they are following, and when they joined the Twitter service.

With the tweets at our disposal, we ran through a number of different metrics to calculate. Some were very obvious and appear in almost any Twitter application: the

[1] http://www.readwriteweb.com/archives/more_twitter_analysis_influencers_dont_retweet.php

number of replies to other people, the number of retweets, hashtag usage, and the sharing of links. We floated around the idea of using some NLP to do sentiment analysis on tweets, to calculate the number of happy and sad tweets. In the end we decided to keep things simple and just use things that could be captured with regular expressions. There were two reasons for this. First, we wanted to keep things simple and clear for the user; the process of calculating the number of “happy” tweets may be interpreted as somewhat mysterious. Second, we wanted to keep the possibility of dynamically adding data to the visualization, so we didn’t want to use anything that would take too long or that would require anything more than JavaScript to process.

We decided on collecting the following data for each user, although we quickly decided that not all of them would have to go into the visualization:

- Twitter username
- Actual name. In some cases the celebrity didn’t actually fill out this field properly, so we manually corrected it.
- Number of followers
- Number of people the celebrity is following
- Total number of tweets
- Number of days on Twitter. This is calculated based on the date they joined the Twitter service.
- Number of tweets per day. This is the total number of tweets divided by number of days on Twitter

The following were all calculated as raw numbers and as percentages (out of either 3000 or all of their tweets, if that number was lower):

- Number of replies—Starting a tweet with '@', in response to another user
- Number of mentions—Distinct from replies, containing '@' somewhere else in the body of the tweet. We thought it would also be interesting to look at how many other users mention a particular celebrity, but the API limits how this could be done. Also, it turns out that celebrities can get a ridiculous number of tweets directed at them; see http://waitingforbieber.com/ for a live feed of users trying to get Justin Bieber to follow them on Twitter.
- Number of retweets—Reposting the content from another user, denoted by the "RT @" syntax. The data for this was actually a bit tricky. Twitter introduced functionality to retweet by the press of a button; however for backwards compatibility reasons, retweets generated this way are stripped from the user timeline available from the API. Retweets produced without the button were found by using a regular expression, retweets produced using Twitter's functionality were calculated by subtracting the actual number of tweets returned by the API from the expected number of tweets returned (usually 3000, possibly lower).
- Number of tweets with a hashtag—The '#' syntax is used to denote a topic that a user wants to make searchable. We thought about looking at the actual hashtags used, but that was too fine-grained.
- Number of tweets with a link

- Number of tweets with a link to a photo—Based on looking for specific image hosting sites such as TwitPic, yFrog, Flickr, etc.
- Number of tweets with an emoticon—Based on a pretty broad regular expression
- Number of tweets with multiple exclamation points—We wanted a metric that would try to capture some sort of emotion, like excitement. We also thought about looking at tweets that featured words entirely in capital letters, but things like the 'RT' syntax would throw that off.

We also manually added some data for each user that was not available from the API, but we felt would be useful to filter by:

- Occupation—In our data set: actor, musician, athlete, and politician. Some celebrities fell under multiple categories.
- Gender

Scanning the data itself, we ran into some issues. A number of celebrities had fewer than 3000 total tweets, so we felt that the raw counts would be misleading. Instead we decided to use percentages in the visualization, although the raw numbers would be available if needed.

The ranges for each metric varied wildly. For the sake of data integrity and not misleading the user, it might make sense to show the entire range, from 0 to 100%. However, this would lead to bunching and crowding for most of the metrics. For example, John Boehner uses the most hashtags out of anyone in our dataset, but that maximum value is only at 27%. Displaying the entire range from 0 to 100% would look very empty and would not be a good use of space. Also, we felt that the relative values were more informative than absolutes. By itself, 27% might not seem like that much, but it is in fact the highest percentage of hashtag usage within the entire data set. Therefore our axes are scaled to show the range of values that are present within the data set.

Another feature of the data is the fact that there are a quite a few outliers. We show an orange line denoting the average celebrity. If you examine this average line in the visualization, it stays within the bottom third of the window. Although it is always interesting to look at the values of these outliers, it can be less than desirable if you are trying to explore the majority of the data. A lot of the data is densely packed toward the bottom (the presence of outliers prevents the axis from being scaled such that the focus is on the area where most of the data is). For example, Donnie Wahlberg has by far the highest output, nearly 61 tweets per day. Wyclef John is the next highest at 27 tweets per day. However, most people just tweet between once and five times a day and these outliers prevent us from seeing that distribution more clearly. The pull of these large outliers also makes it difficult to see outliers in the opposite direction. There are a couple of people who tweet less than once a day but they all just get collapsed at the very bottom. This is really just a property of the data set though, and cannot really be designed around.

Design Decisions and Iterative Process

Parallel Coordinates

The foremost goal of our visualization was to delineate the "shape" of Twitter success for individual celebrities in a comparable and filterable format. We aimed to form this success shape based on a celebrity's behavioral (e.g. replies, retweets) temporal (e.g. tweets per day), and relational (e.g. followers, following) history.

From our project's conception, we realized we would be working with a multivariate data set and would need a clear and manageable way of encoding and communicating it. Hence, we decided parallel coordinates would be an apt visualization technique to create a high-level success shape by connecting data points. When a user scrolls over a line, that particular celebrity's shape will be brought into focus.

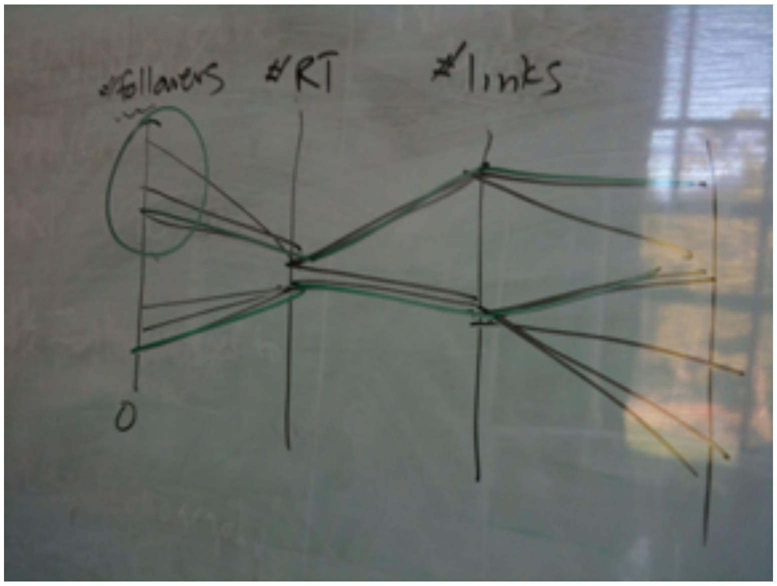


Figure 1: Initial conception of parallel coordinates visualization

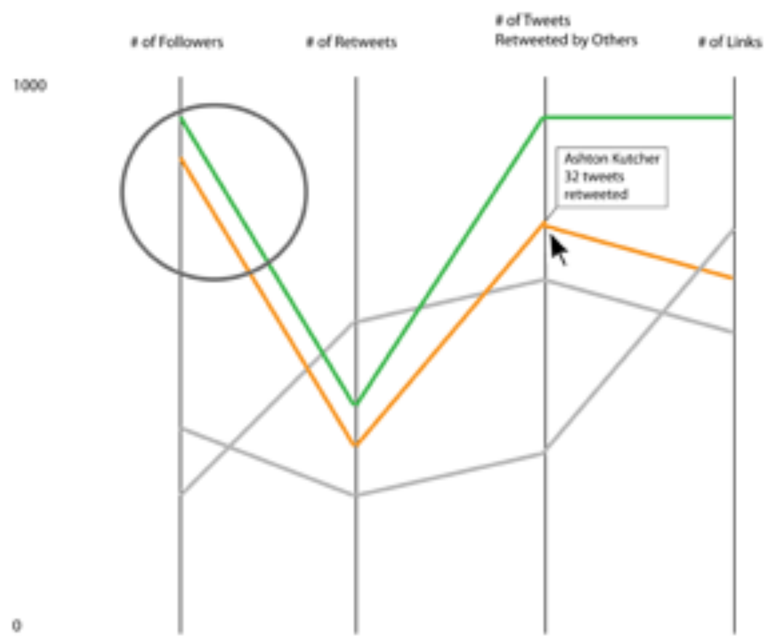


Figure 2: First low fidelity mockup with selection, brushing, and tooltips

Parallel coordinates give us several ways to encode and manipulate multidimensional data; having distinct ways to filter on several axes allows for the picturing of many different stories. We particularly feel that the parallel coordinates implementation in Protovis lends itself elegantly to filtering by value; by resizing the axes, users can highlight the success lines of celebrities whose data points fall in the appropriate ranges. We decided to supplement this filtering mechanism with a set of controls to filter on non-axis based, nominal criteria. Moreover, we added brushing for the lines that are hovered over in order to highlight the shape of particular celebrities.

We used the Protovis library to create the main parallel coordinates plot. Protovis was appealing because not only is it open source, but it is built on javascript, which we were all familiar with. We had also considered using Flex, for which someone had already developed code to create a parallel coordinates plot (http://www.michaelvandaniker.com/labs/cerealPCP/CerealPCP.html), but we found that

Protovis was more fully featured. Protovis also has an implementation of parallel coordinates (http://vis.stanford.edu/protovis/ex/cars.html), which uses json objects to store the data and allows the user to filter by ranges on each axis. This was a good starting off point, but it lacked a lot of the interactivity that we had envisioned, so we would have to add that ourselves.

Much of the interaction was handled by using jQuery and building on top of the Protovis code. We had trouble getting around some of the limitations of the underlying Protovis code, for example the dragging and rescaling action on each axis is a bit unintuitive and it's not readily apparent that the axes are interactive at all. Hit detection is also a bit fuzzy, staying too close to an axis will not highlight a line (the cursor changes to crosshairs when this happens). The placement and formatting of the labels was also somewhat out of our hands.

The first interaction we added was the ability to highlight individual lines. This is crucial in a parallel coordinates plot because the visualization can appear very cluttered. It took a while to figure out exactly how Protovis was manipulating the data, as the documentation is still a work in progress. Eventually we did figure out how to get this interaction to work and we played around a bit with how a line should appear highlighted. We decided on both changing the color (to a darker blue) and making the line a bit bolder. All of this is handled within Protovis. Highlighting a line in turn brings up the details for the twitter user that the line represents (Javascript/jQuery).

Axes, Lines, and Filters

<u>Choice of Axes</u>

In our subsequent designs, we attempted to split the dataset on the basis of "independent" and "dependent" variables. We initially decided that the axes for the parallel coordinates would reflect the independent variables corresponding to a Twitter user's behavior (i.e. percentages of tweets containing replies, mentions, retweets, hashtags, links, photos, emoticons, and multiple exclamation marks). Correspondingly, we decided that while a line through these data points would reflect a behavioral pattern, we could vary the width of the line to encode the number of followers.

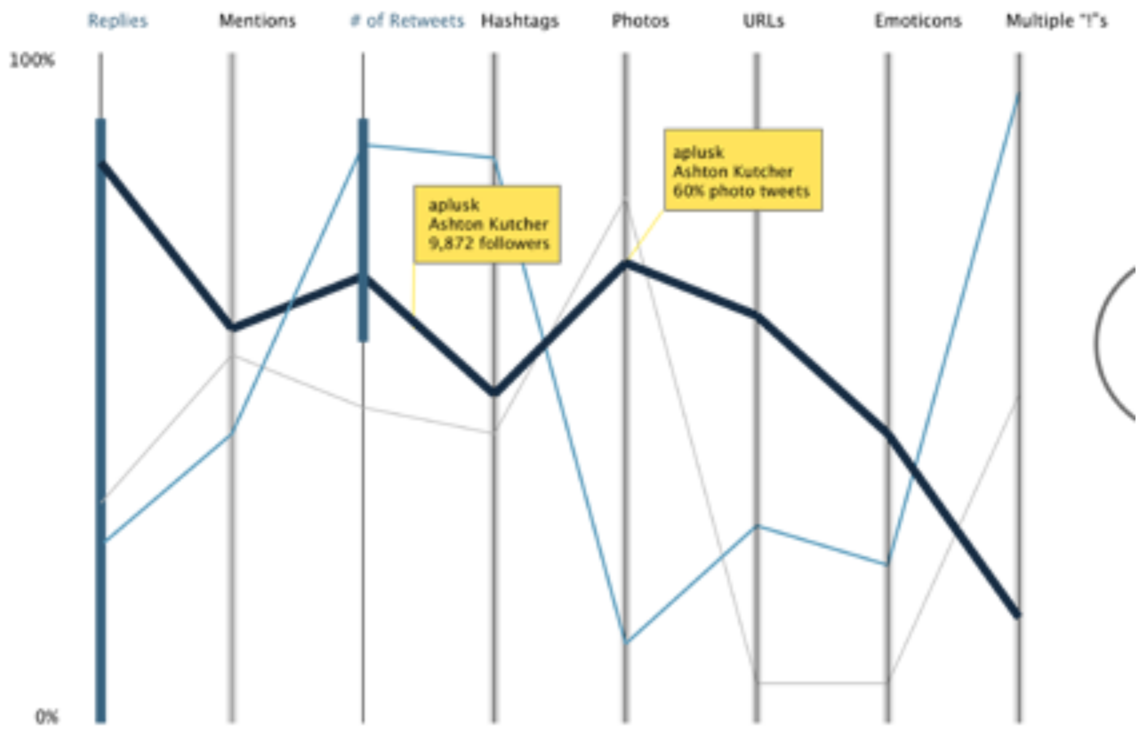


Figure 3: Subsequent low fidelity mockup with axes-based filtering, gradational coloring, differing line widths, and tooltips

We soon realized, however, that using line width to indicate number of followers was not feasible, as not only was it not precise enough, it would also result in occlusion problems with thicker lines covering thinner ones. Hence, we devised two alternate methods of integrating followers—as a set of filter buttons or as an axis. From our user testing, we found that users across the board preferred having followers as an axis because it allowed them to make distinctions between celebrities. Based on our user studies, we also decided to include the celebrity's average tweets per day as an axis as well, in order to depict how prolific a celebrity is in conjunction with his or her followers and behavior.

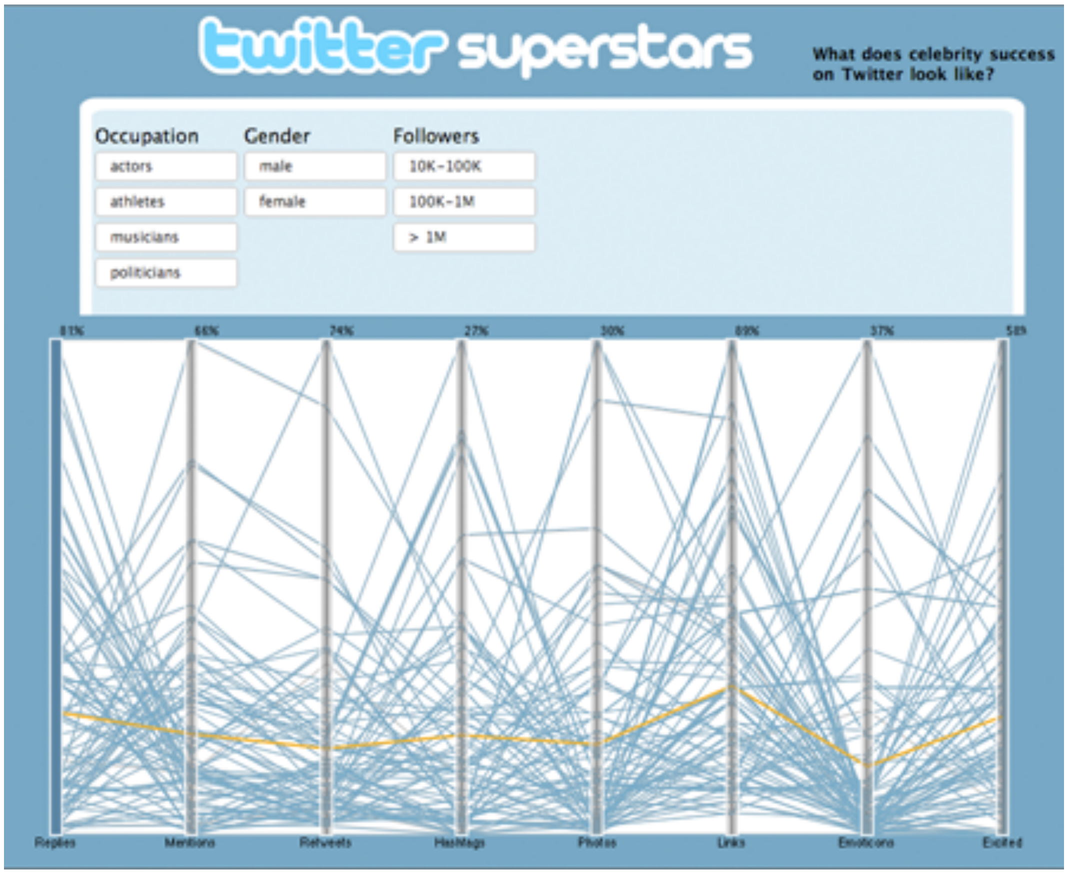


Figure 4: Visualization with followers as buttons presented for user testing

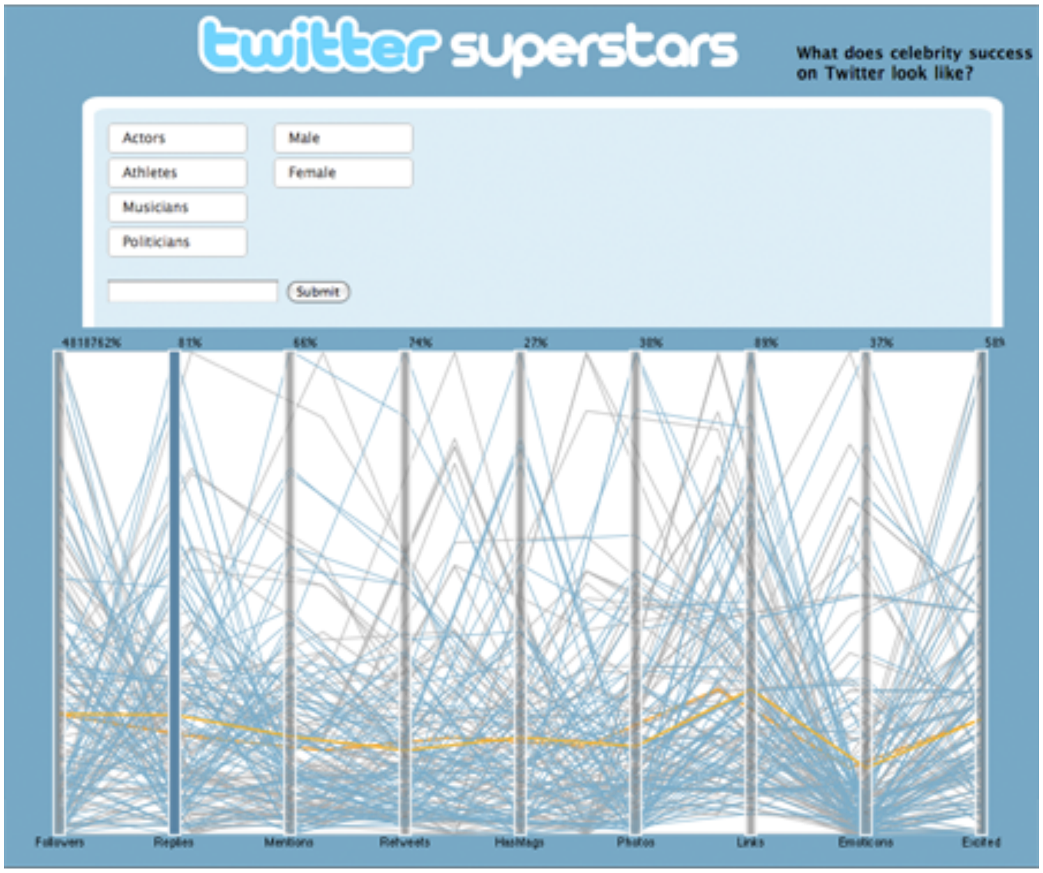


Figure 5: Visualization with followers as an axis presented for user testing

Hence, by making followers an axis, we eschewed the idea of a representational distinction between independent/dependent variables and behavior/success. In fact, users found that the shape of success was more obvious when the success measures were placed alongside behavioral data.

In the course of our user studies, we also found that not all of the behavioral data was useful or appealing to users as parallel coordinates, so we decided to remove these axes. For instance, users did not immediately know the difference between replies and mentions, which both indicate usage of the @ symbol (to clarify, a reply is a direct response to a user where the @ reference appears at the start of a tweet, whereas a mention is a reference to another user elsewhere in a tweet). Similarly, since users found that the term “excited” was ambiguous, and it was one of the less important measures, we decided to get rid of that axis as well. Furthermore, having more than eight axes was unwieldy, so we decided upon a final list that included followers, tweets per day, replies, retweets, hashtags, links, and emoticons.

Ordering of Axes

During the period where we only represented behavior across the axes, we decided to group behavior by Twitter usage. We placed replies and mentions next to each other since they were variations on using the @ symbol. We followed these by retweets and hashtags, as the former is a Twitter operation and the later a Twitter convention. Since hashtags lead into content, we decided to place photos and links next to hashtags, mentally grouping the three as “media”-based behavior. We followed these by emotional behavior in the form of emoticons and exclamations.

When we moved away from the behavioral model and decided to include followers and tweets per day, we included these at the leftmost axes. Since users were most interested in these axes to determine the holistic success levels of celebrities, we perceived these as the most important and therefore deserving of leftmost status.

Labeling of Axes

With respect to selecting particular fields from our dataset and labeling the axes of the parallel coordinates, we realized early on that using percentage-based data rather than numerical data would enable fair comparisons, since data becomes skewed when it is based on the sheer volume of certain celebrities’ Twitter activity and membership time. We also found that it was not feasible to let all the axes be on a conventional scale of 0–100% because the data was often grouped in a small chunk at the bottom of the axis, which becomes unreadable and undistinguishable. Hence, we decided each axis would be scaled from 0 to the maximum value in order to spread out the data points. This meant that labeling each axis at the top with a percent value was critical. When we decided to revamp the axes and include followers and tweets per day, we were also dealing with different units of data, and we had to make it clear that only the behavioral axes were percentage-based.

Filters & Celebrity List

We decided that in addition to the built in filtering by axis of the parallel coordinates, we wanted to include more high-level filters to group celebrities based on qualitative and quantitative criteria. Hence, we drafted a list of possible filters—occupation, gender, number of followers, number following, number of tweets, tweets per day, and time on Twitter.

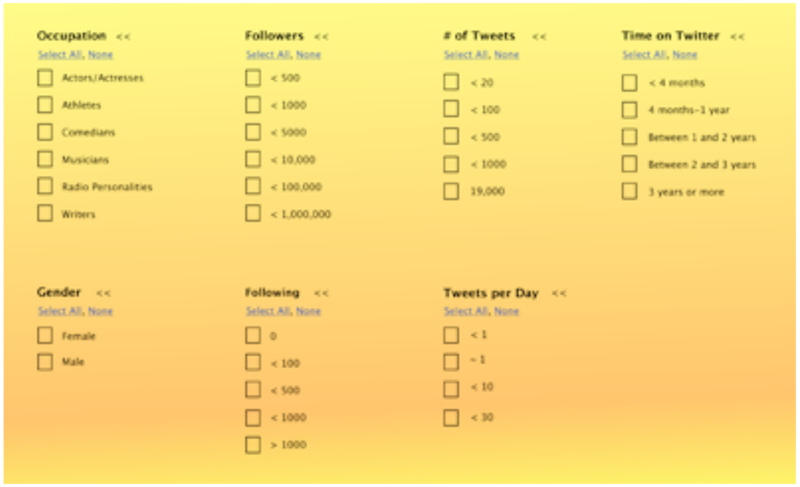


Figure 6: Original ideas for filterable criteria

The Protovis documentation was less than clear in explaining how we could implement filtering. Eventually we found that we could use JavaScript to manipulate the underlying array of data and then have Protovis redraw the entire visualization. The entire process appears quite seamless. We did not want to completely remove the filtered out data from the visualization, since comparisons may still be made with the rest of the data set. Instead, filtered out data remains on the visualization but appears greyed out in light grey. This was actually accomplished by using an image of the entire visualization greyed out as the background image. Removing a line from the visualization by filtering reveals the underlying grey line.

As we spoke to users and changed the axes of our parallel coordinates, we narrowed down the list of filters based on what was interesting. We first decided on occupation, gender, followers, and tweets per day to categorize at identity- and success- based levels. However, when we decided to encode the latter two in the parallel coordinates, we decided to go just with occupation and gender.

From our user testing, we discovered that having a list of filters was not sufficient—often users were interested not in types of celebrities, but a particular celebrity. Hence, we decided to include a list of celebrities so that users could pick and choose whom they wanted to focus on. This was a critical change because it enabled the reproduction of results without forcing a user to rely on their memory of what a particular celebrity's line looked like.

The list itself is linked to the data in the visualization; mousing over a name in the list highlights the celebrity's line in the visualization, and the reverse happens when a line is highlighted. The names of the celebrities that are currently filtered out of the visualization are also greyed out, although they can still be moused over to bring up their details on demand.

The interactive list of names is largely implemented in JavaScript and jQuery. When a name is moused over, a flag is set and Protovis redraws the entire visualization with that line highlighted. Mousing over unsets the flag and the visualization is redrawn without that line highlighted. Even with all the redrawing, there is not much lag at all. A similar flag-setting process is used when a filter button is clicked and the algorithm is calculating which lines should be displayed.

Color

We wanted to visualization to immediately resonate “Twitter,” so we decided to mimic their color scheme, using teal, orange, light blue, and white. However, we also wanted to use color to meaningfully encode information. Our first idea was to use a gradient, where, depending on the particular axis selected, the lines that appeared toward the top of the axis (i.e. those with higher values on that axis) would have a higher saturation as opposed to those that fell lower. We soon eliminated this idea, though, because it did not seem necessary to doubly encode the information, and it might not be immediately comprehensible for a user to see the gradient change across axes when manipulating them.

Based on the feedback we received from the class in our midterm project presentation, we decided to attempt to encode occupation information via color. We found that this was aesthetically unappealing (it was in fact alarmingly chaotic) and did not do much for the visualization with respect to highlighting patterns across occupation.

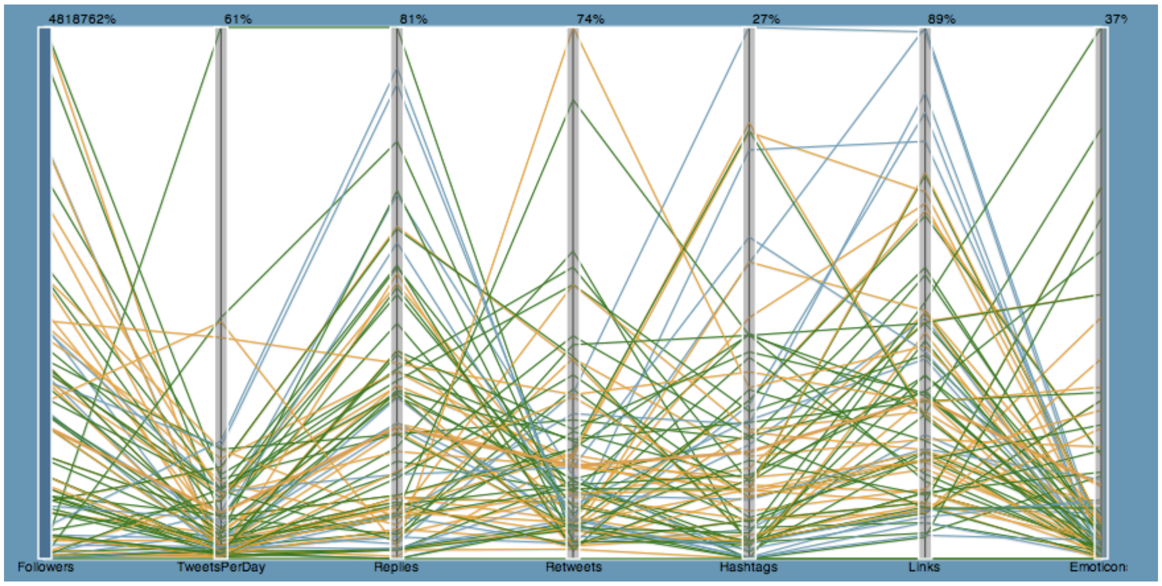


Figure 7: Ineffective and potentially alarming use of color for encoding occupation

We ultimately found a usage for color as we added more functionality to our visualization—we decided to distinguish the average and newly inputted lines (discussed below) with color.

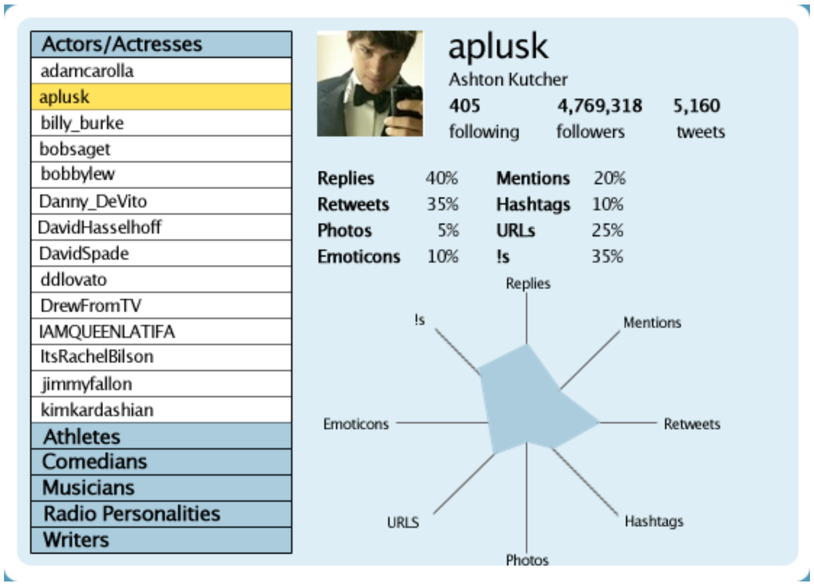
Details on Demand

We realized that even though parallel coordinates enable us to show the general shape of behavior, it is not conducive to depicting specific values. We decided at first to use tooltips to display the actual values. We soon found, however, that the functioning of the tooltip may be inconsistent. For example, if a user were to hover over a particular intersection point on an axis, say replies, the tooltip should display the percentage of replies for that celebrity. However, if a user were to hover over a section of the line in between two axes, what information should be displayed? Moreover, we felt limited by the size of a typical tooltip, which can only display a few lines of information compactly.

A tooltip can also obscure a lot of the line data, depending on where it is positioned.

We decided to replace tooltips with a details-on-demand panel, which reflects all the relevant numerical information about a celebrity's line when hovered over. Not only does this display the actual values for each axis, but it also displays other relevant and interesting data that we have collected. We also decided for aesthetic value to style this section to mirror a Twitter profile—including the celebrity's picture, twitter handle, real name, followers, following, and tweets.

One of our early ideas was to include a starplot of a celebrity's behavioral shape; however, we realized that this was not only redundant, but also a bit difficult, as celebrities would often have zero values which would eliminate an axis altogether.



Actors/Actresses
- adamcarolla
- aplusk
- billy_burke
- bobsaget
- bobbylew
- Danny_DeVito
- DavidHasselhoff
- DavidSpade
- ddlovato
- DrewFromTV
- IAMQUEENLATIFA
- ItsRachelBilson
- jimmyfallon
- kimkardashian

Athletes
Comedians
Musicians
Radio Personalities
Writers

aplusk
Ashton Kutcher

405	4,769,318	5,160
following	followers	tweets

Replies	40%	**Mentions**	20%
Retweets	35%	**Hashtags**	10%
Photos	5%	**URLs**	25%
Emoticons	10%	**!s**	35%

Figure 8: Initial idea for details on demand with actual collected values and starplot (later eschewed)

Average Line

In later versions of our visualization, we decided to add an average line, in order to give some context and enable comparison when examining any celebrity's line. We decided to make this line orange (the complimentary color to Twitter's teal) so that it would be omnipresent and beg to be examined outright. Furthermore, we decided that the details on demand would default to the average values and show an orange Twitter bird so that it would be clear what the line referred to.

Add Yourself Functionality

We hypothesized that having the ability to add your own Twitter success line to our visualization would make it more interactively compelling and decided to include it in later versions. We found that our supposition to be supported by user testing. In order to differentiate new lines added to the visualization, we decided to display them in a new color. One problem that we encountered early on was the lack of feedback—when a user added himself or herself, the progress was slow and sensitive based on Internet connectivity, and users would be unsure whether it was even working. Therefore, we decided to include a progress bar to chart the status.

We implemented this feature by using a series of Ajax calls to the Twitter API, retrieving user data and their most recent 1000 tweets. JavaScript is then used to calculate the same statistics featured in the rest of the data set, and then the data is added to the visualization and redrawn by Protovis. The Ajax code is very sensitive to the speed of the network, and sometimes gets stuck getting a portion of the data (hitting the "add yourself" button again helps). Refinements should probably be made to its efficiency, but for now it works.

User Testing

We tested four Twitter users. Three of these users follow celebrities. There is quite a range in the number of celebrities followed, from Shaquille O'Neal to Conan O'Brain to Neal Patrick Harris to Sockington the cat. We gave each user a brief introduction to the purpose of our visualization, stating that we consider celebrities to be successful on Twitter if they have a large number of followers and a high level of activity. We told them we had two alternate visualizations and had them tell us when they were ready to try the second one. The first visualization had the number of followers as filter buttons. The second visualization had the number of followers as an axis and included the Submit functionality for adding another line for any Twitter user. We had them play with both visualizations while thinking out loud. Each test lasted between 10 and 15 minutes long. We observed and took notes on what they were doing and saying.

Parallel Coordinates

All of the users found themselves drawn to playing with the parallel coordinates. They did not investigate the filter buttons until later in the test, and professed to enjoy the parallel coordinates more than both the filter buttons and details on demand. However, it was not evident to them how to use the parallel coordinates. They spent the first portion of the tests hovering over lines and commenting on their shapes. One user accidentally discovered that the axes could be dragged when she clicked on one. As for the others, we told them they could interact with the axes and gently guided them in the right direction when they still could not figure out how to manipulate the axes. It was clear that the axes did not look draggable. Only one user figured out how to shrink and drag the axes. He tried different scenarios, decreasing Hashtag usage, increasing Replies, and increasing Emoticons. He discovered while doing this that the orange average line is no longer clickable when it falls above the axis' height. The users also did not know how to reset the axes. One user suggested adding a Reset Axes button.

Use Behavior

As mentioned before, users hovered over lines and commented on their shapes. The three users who follow celebrities behaved differently from the user who does not follow celebrities. This user made broader observations, for example, that males and politicians love links and females use lots of emoticons. The other users either hovered over lines and expressed surprise or agreement with the shape of a particular celebrity's line ("Obama doesn't use photos? How bizarre.", "Huh, David Hasselhoff is really connected to his fans. He replies a lot.", and "Are these celebrities broadcasting themselves?") or made guesses as to what a celebrity's line might look like. One user analyzed Terrell Owen's line, and noticed that his tweet structure suggested a conversational nature, whereby he replies to his followers. Another user who had hovered over Britney Spear's line earlier wanted to find it again but couldn't remember its full shape. She remembered that Britney Spears had a high Excited percentage and no Emoticons. She used this method to find the line again. The user who analyzed Terrell Owen's line also attempted to find celebrities' lines again after hovering over them. This suggested to us that users would appreciate a way to quickly find celebrities' lines that is faster than the Submit functionality. The user above recommended that we add a clickable, alphabetical list of celebrities.

Followers and Tweets per Day

We asked users after they had tried both visualizations whether they preferred seeing the number of followers as an axis or filter button. All four users chose the former. One user elaborated that the number of followers is a big number, and therefore more suited to an axis. We then asked the users to tell us what other information they would like to be able to add to the shape of the lines. Three of the users suggested tweets per day of their own accord, even though we had a list consisting of tweets per day, time on Twitter, and total tweets to ask about if they did not suggest anything. One user told us he would like to be able to filter by the change in behavior over time. He gave us an example: “How has tweets per day changed over the last time frame? In weeks and months.” Another user suggested that tweets per day would be an interesting statistic. When we asked about adding time on Twitter and total tweets, she said the time on Twitter would be useful but total tweets would not be so useful. The third user who suggested tweets per day asked us to add it as an axis after the number of followers axis. Another user cast her vote for tweets per day as an extra filter, and told us that she sees time on Twitter and total tweets as being roughly the same. The fourth user explained that tracking total tweets is not fair because it’s affected by how long a user has been on Twitter. He dismissed the time on Twitter as being uninteresting.

Adding a Line

The Submit functionality for adding a new line for any Twitter user needed some refining. Since it did not come with any hints or instructions, the first user searched for “ashton”, which did not come up with any results. We learned to add an instruction to use a person’s Twitter handle. The other glitch was that users hit Enter rather than clicking on the Submit button. Also, the Submit button was not self explanatory and was perceived as a vague search tool. We later changed it to Add Yourself to dispel that ambiguity. When the next user saw the Submit button, she commented that she wanted to browse rather than search. She searched for a few celebrities, but both lines came up red, so she could not quickly distinguish the second celebrity from the first one. The third user did not initially know the purpose of the Submit box. We realized while explaining it to her that in addition to including an instruction about using the Twitter handle to add a person’s line, we should also instruct that the account needs to be public. When she typed in her Twitter handle and saw her line come up, she exclaimed, “Oh my god, is that me, that red line??” She was very excited, and said, “That’s cool. That’s cool!” She told us that she would like to see if she tweets like Jennifer Aniston. The last user guessed that the Submit button required a celebrity Twitter handle or a term. He was very interested in seeing his line as well.

Filter Buttons

Beyond the new Submit and Followers as a filter features, our filter buttons behaved contrary to how users expected. We had four different occupation buttons and two gender buttons. All were selected by default, but whenever a user tried to see only Actors, he or she had to deselect the other three occupations. This caused some confusion, which we later fixed by adding All Occupations and Both Genders buttons that become unselected when any other button in the same category is selected.

Meaning of Axes

All of the users asked what Excited meant. It was listed as “!s” in the details on demand. The first user asked us whether we also included tweets that had all capital letters. Upon further discussion, we concluded that analyzing tweets for capital letter usage is a non-trivial task. We also eventually decided to remove Excited as an axis to make room for more salient axes such as Followers and Tweets per Day. The Mentions axis was also a point of confusion for some users. They assumed that it referred to the percentage of other people’s tweets that mentioned the celebrities. This was another axis that we removed. We took it out of both the details on demand and the parallel coordinates because of its similarity with Replies and its ambiguity.

Labeling and Filtering Out

On a minor note, none of the users understood the labeling of the axes. They guessed that they were percentages of total tweets that had, for example, 37% emoticons. Also, the lines that were filtered out were too dark of a grey, so some of the users did not notice when lines became greyed out. One user told us she preferred color coding. All users had difficulty precisely targeting the lines.

Future Work

Although we have a working model that allows users to explore the data we extracted from Twitter, we still have some goals that we would like to complete.

Different average statistics can be presented to the user. For example, when the data has been filtered to just male athletes, a new line displaying the average statistics for those celebrities should be displayed. We wrote code to calculate average statistics;

however it does not cover the case where the data set has been filtered by a range of values. The Protovis code handling range filtering is still somewhat mysterious and undocumented. The display of additional user requested lines should also go through some usability testing to determine the best colors for the lines and how they should appear in the visualization.

We would like to include the ability to view the celebrity's behavior over time. Our current implementation is based off of the last 3000 tweets the celebrity has cached in the Twitter database. Viewing trends over time may suggest a deeper meaning to the data and reveal trends that may be more temporal based, such as actors tweeting more frequently near the release of a movie, or athletes tweeting more often around a competition.

We also had a feature request to show trends by time of day, days, and months. This would be useful to see when celebrities are most likely to tweet, and if there are any relationships between the time of day that a celebrity tweets and other behavior, such as percent of links, hashtags retweets and mentions. This can also act as a proxy for who does the tweeting; for example, if there is a bimodal distribution for one person, it might suggest that they have a publicist also doing their tweeting.

We would also like to include functionality that lets the user compare two or more celebrities in our visualization. The parallel lines provide information about celebrities in relation to each other, but it does not show details comparing celebrities. To see details, the user needs to mouse over either the name of the celebrity or their respective line to get the detailed statistic we generated from their tweets. Currently, to compare celebrities, the user has to use remember the details and shapes of the lines.

We also agree that showing celebrity friend connections is important. This could lead to exploration of how celebrities communicate with each other through Twitter, if they go back and forth replying, or if they mention only friends or strangers. This could show if celebrities use Twitter to socialize with other celebrities, or if it serves more as a broadcast tool to their fans.

Our last idea is to research how terms are used in Twitter. This would require more energy on our part, as proper natural language processing scripts would need to be written to determine the meaning and structure of tweets. This could show what content is discussed in tweets that are broadcasted by celebrities: whether they are quips, advertisements, or real conversations.

Conclusions

The TwitterSuperstars visualization provides an engaging way to view successful celebrities. It is not enough to be famous, or have the highest number of tweets per day. Our visualization gives a holistic view of all the aspects that make up a Twitter user's behavior, such as hashtag usage and links to interesting content. Deciding on which structural characteristics to feature in the visualization was tough. We limited the number of filters to show only the most meaningful information to the user. The development process went fairly smoothly. While the documentation for Protovis is largely lacking, starting with the parallel coordinate demo from the Protovis website helped out greatly.

In the process of developing the parallel line coordinates, we considered different ways to display the information, such as adding tooltips to show information on demand, labeling the lines with the associated values as they cross the vertical filters, and having buttons to add filtering functionality. Our user testing gave us useful feedback on the design. We were able to mediate users' difficulty with parallel coordinates by providing short, concise instructions. Our users enjoyed using our visualization and gave positive feedback about their experiences.

Our process had us moving away from defining what success looks like, towards developing a visualization that lets users explore and play with the data we had on celebrity tweets. We think there is great value in letting users conceptualize celebrity twitter statistics in a way that is fun and revealing about the underlying data. To explore the visualization, visit http://bit.ly/twitterSuperstars.