Gathering and dynamic visualization of time-based changes to data generated by Reddit.com

Abstract

With the increased amount of data generated by social networking sites there is also increased difficulty in the analysis of that data. Time-based changes to data may provide insights into the effects of popularity in social networks. Information visualization may be a vital tool to assist social scientists with analysis of large quantities of data. However the gathering and formatting of time-related data from social networking sites still remains an obstacle. This project explores the process of gathering time-based data in real time and using dynamic visualization techniques to visualize and analyze time-based changes in data generated by discussions on the social networking site Reddit.

Introduction

Information visualization systems and components

Information visualization is the process of creating visual representation of an abstract set of data through the usage of conventional or computer graphics. The goal of information visualization is to assist end users in perceiving and analyzing the underlying patterns and effects that exist within the data set that may otherwise be difficult to recognize. The visualizations created from the data set assists analysis through the addition of another dimension to the data beyond the alphanumerical dimension, typically by displaying the relationship between different units of data within a set or the relationship between different sets of data.

According to information visualization theory, there are seven components that are common to all information visualization systems. The inclusion and implementation of these components are necessary in order to make the information visualization system effective at providing and communicating information to the end user (Luse, Scheibe &Townsend, 2008).

These components include:

Overview: A method to show the end user the entirety of the data set from which the visualization is created, providing a sense of structure and relationships that may be present in the visualization.

Zoom: A function that limits the scope and scale of the visualization in order to focus on areas of the visualization that end users are analyzing or find interesting, a well implemented zooming function should allow the end user to retain a sense of relative position within the overview.

Filter: A function that removes or highlights only subsets of the information within the visualization as defined by the end user in order to draw attention to interesting data or away from unimportant data.

Details: Detail regarding the data set from which the visualization is created should be easy to access and readily available per the request of the end user.

Relations: Relations between the data from which the visualization is created are often as important, if not more important than, individual data values and is a primary

reason for the creation of visualizations. A visualization system should have clear visual indications of the relationships between data or data sets.

History: Time based changes to the underlying data set of the visualization may often provide context and additional information regarding the data or sets of data that are otherwise difficult to see in non-visualized or static forms.

Extraction: End users should be able to extract necessary information or portions of the visualization for additional analysis, future review and display.

Social networks in the internet era

A social network is, in the simplest terms, the connections that are formed by the social interactions of a group of individuals. More specifically, a social network is a group of people who may form personal relationships, or socially interact with one another on regular basis (Wasserman, Faust, 1994). Traditional social networks are formed geographically and generally remain localized. The advent of the internet has significantly changed the social spaces within which people are active and has redefined the concept of social networks (M. Christensen, Jansson, & C. Christensen, 2011). The term social network has begun to be more commonly used to refer to people connected through the use of computer communications (Garton, Haythornthwaite, & Wellman, 1997).

Social networking sites are the localization, organization and facilitation of social activities in the online realm, often sporting a formal and organized format that makes social connections visible to the public. These social networking sites grew from the more informal

and less organized systems of communication services such as internet relay chat and blogging services. In the modern day, each of these social networking sites prioritizes and orients itself towards a specific context of social interaction, usually delineated by goals, interests or activity (Boyd, Ellison, 2007)

Social network analysis is a subcategory within the social sciences that focuses on the study of interactions between individuals within a social circle or social network utilizing mathematical techniques and metrics native to graph and network analysis (Scott, 2012). With the widespread adaptation of social networking sites in the last few decades there has been a dramatic increase in the interest in using social network analysis to process the large amount of data that has become available to social scientists (Serrat, 2010).

Visualization in social network analysis

While there has been an increase in the amount of data available to social scientists, the gathering and organizing of this large amount of data into usable forms has been a hurdle to large scale social network analysis. The user interface of social networking sites are usually designed to facilitate discussion and interactions between users, and while these interfaces work for their intended purpose, they do not serve well as an information visualization system to assist social network analysis, lacking most of the components that are vital to information visualization systems (Ellison, Steinfield, & Lampe, 2007).

The creation of information visualization systems for social network analysis is also complicated by the differences of the underlying structures of social networking sites. The very thing that social networking sites do in order to separate themselves from their competition

has made it difficult to create a universal tool to gather, analyze and visualize data generated by social networking sites. Information gathering and visualization systems need to be tailored to individual social networking sites. Many such tools have been created to facilitate data gathering and analysis on a number of different popular social networking sites (Brandes, Wagner, 2004), both by joint efforts between social scientists and programmers and by hobbyist programmers driven by curiosity instead of the pursuit of solid results (Garton, Haythornthwaite, & Wellman, 1997).

One such social networking site is Reddit, an online community image and message bulletin board with content submitted and curated by users through the usage of a real time scoring system. While a few visualization tools have been created in order to visualize the data generated by Reddit, none have incorporated dynamic time-based changes as an aspect of social interaction.

Goal of project

With the changes in content consumption habits of individuals through the use of the internet, popularity has become a very important metric for measuring the success of produced content, and dynamic time-based changes to consumption and sharing of content can impact end-state popularity. Using Reddit as a platform, the goal of the Real Time Conversation project is to create a visualization system that can view the dynamic time-based changes of social interaction that takes place on Reddit.

Overview of Reddit

Reddit is an online community image and message bulletin board that hosts content submitted and curated by its users. Reddit's unique structure allows users to cast votes on the submitted content using a democratic system of one up or down vote per user. From the perspective of an individual user, upvoting submitted content is to judge that content to be likeable, good and interesting, whereas downvoting the submitted content is to evaluate that content as unlikable, low quality or repulsive. The aggregate effect of this rapid fire democracy is to determine, via input from the user base, what content submitted to Reddit can be considered constructive and beneficial to the community as a whole and what content is considered to be disruptive and destructive to the community. The same democratic voting process is also extended to the individual comments within the discussion section of Reddit, where users discuss and comment on the submitted content.

The majority of the content sharing and discussion activities on Reddit take place in user created sub-communities known as Subreddits. These Subreddits are often delineated by and focused on specific topics or themes. Some Subreddits focus on sharing content and having discussions over a specific subject such as technology, politics or latest popular television show, other Subreddits may be for sharing content in a specific format, such as discussions, videos or animated gifs. Because of the ability of users to create Subreddits, one of Reddit's strengths as a social networking site is that there is a Subreddit for just about every topic. Of course, the activities and popularity of these Subreddits do still rely on the level of user activity within.

Reddit selects a choice subset of Subreddit that they consider to either be interesting and relevant to current events or active and well moderated to make up the default landing page for Reddit.com. This landing page, known as the Front Page, displays a list of twenty of the most popular content submissions from the subset of Subreddits, allowing casual users or unregistered readers a low commitment way of engaging with the content available on Reddit. Content with the highest aggregated score within the choice subset of Subreddits is displayed towards the top of the Front Page, allowing it to gain a great amount of visibility and, in turn, additional popularity and score. Content that makes it to the Front Page is periodically moved down the list in a process independent from voting in order to allow room for fresh new content. The tiered process for displaying content on Reddit works synergistically with the democratic voting process. Content that is considered “better” gains more upvotes and becomes more visible, where it is capable of gaining even more upvotes and more visibility until it has run its course. The best content within a period of time is capable of picking up the most traction from this positive feedback loop, with the greatest amount of traction being gained when it reaches the Front Page. On the other hand, low quality or bad content becomes less visible as they are downvoted, making it harder for any other users to view, and more importantly, vote on the content, thus causing it to sink into obscurity.

In addition to the ability to aggregate high quality content through the democratic voting process, Reddit is also host to discussions regarding the submitted content, with certain Subreddits such as AskReddit, askscience, explainlikeimfive and IAmA being solely dedicated to discussions. In these discussion focused Subreddits the original submitted content, usually in the form of a question, is treated as a conversation starter and the discussion makes up the real

substance of interest. The discussion section is also subjected to the same democratic voting process that governs the submitted content, allowing comments to be upvoted or downvoted by the community, creating a similar aggregate effect within the discussion. Likewise, the individual comments of the discussion are also sorted and displayed in a manner similar to the submitted content, with a vertically nested list of comments that contain a list of its child comments and so forth. After a certain number of nested comments, Reddit puts a link to the additional discussions in place of another round of child comments, keeping the threads from becoming too long or losing focus.

While there are a few methods to sort the list of comments in the discussion, Reddit, by default, sorts the comments in a manner that promotes discussion, with a mixture of score and number of child comments determining the position of that comment within the nested list, the same process affects each group of nested child comments. Eventually, a reader of the discussion section is left with is a distilled version of discussions that have taken place in relation to the original submission. These distilled discussions are usually informative, entertaining or humorous, and they should be, since hundreds of other people have already evaluated these comments as being such and sorted them through the voting process. Discussions do not remain solely in their respective discussion sections, however. As users move across different Subreddits and different discussions, they bring humorous anecdotes and stories with them to share within another discussion. Similarly, content that is submitted to a Subreddit not within the choice subset of Subreddits that make up the front page may oftentimes be resubmitted on a different Subreddit where, hopefully, more people would be capable of seeing them with the ultimate goal of reaching the front page.

Lastly, individual users on Reddit have a cumulative score known as Karma, a sort of reputation. There are two types of Karma, one for submitting content and one for engaging in discussions, but they function in the same manner, being the cumulative score of all the content or comments that a user has submitted. The Karma system is a sort of encouragement for users on Reddit to remain respectful and follow community guidelines with the threat of losing credibility and reputation.

Current visualizations

Andrei Kashcha of Yasiv.com created a visualization of Reddit in 2012 that shows the relationship between individual Subreddits. Using hyperlinks that point to other Subreddits communities that are contained in any individual Subreddit's description, the visualization created a network graph of the communities of Reddit. This visualization allowed viewers to see the connections between different Reddit communities that may or may not share some common theme. Andrei Kashcha's visualization contains roughly 3800 Subreddit communities and 14000 connections between them, but does not include any of the large number of isolated communities that are unconnected to others (Kashcha, 2012).

Similarly, Randal S. Olson of the University of Pennsylvania's Institute for Biomedical Informatics also created a visualization of the Reddit sub-communities in order to explore how a network map of primary topics of interest may assist users in finding and organizing into specific interest groups (Olson, Neal, 2015).

In the same vein, in 2012, user Laurel Quade created a fictional map of the major communities of Reddit grouping the major areas of interest with the cities scaled to the number

of subscribed users. While not the most accurate or technical of visualization, it does demonstrate the vast variety present in the Reddit community and different entertaining methods of visualization (Quade, 2013).

Kawandeep Virdee of Embed.ly created a visualization of discussions on Reddit in a networks graph in 2014. The visualization created a color coded network graph of the comments of users that participated in any particular discussion, showing both how and where in a discussion users posts, and how often users engage in back and forth discussion with others by posting more than once. The visualization is publically accessible on Github and is an excellent tool to analyze the different structures of conversations that occur in different Subreddits. The visualization highlighted the differences in discussion structures between regular and specialized Subreddits. Specific Subreddits, such as IAMA where users share their profession and answer questions posed by the community, were naturally more prone to reciprocative discussions than other Subreddits (Virdee, 2014).

Visualization of changes in popularity of content on Reddit.com

Reddit has recently taken steps to reduce outside influences to the voting process by making the score of submitted content and comments unknown for a short period of time after submission. This new policy, coupled with the fact that a post and content have their scores displayed as a cumulative number instead of separate upvotes and downvotes mean that regular browsing and static visualizations do not or are not capable of detailing time based changes in the score of discussions. While this is an insignificant problem that does not interfere with the common usage of Reddit, it does make analysis of the effects of popularity in

a discussion inaccurate. A simplistic solution to the lack of time based changes to data may be to assume a linear increase in comment score over the period of existence of a particular post or comment. However, assuming a linear increase in comment score fails to take into account any variances in score that may be caused by opposing reactions in conversations over time or any effects of popularity that may boost the content's score. The solution to the lack of readily available data regarding time based changes to the score on Reddit would be to gather data over a period of real time and integrate that additional dimension of data into current visualizations.

To that end, the Real Time Conversation project aimed to create a solution to the lack of data regarding time based changes to the score on Reddit by recording the data in real time. The end product of the Real Time Conversation project application gathers Reddit threads from the rising page and recursively parses the information related to the submission and comments, with the major variable of interest being score. After twelve hours, the threads that made it to the front page are exported as an undirected relational network in a Gexf file for visualization within Gephi. Each node of the network graph represents a comment and the relation between the nodes represents the nesting of comments within the discussion. Gephi is capable of organizing the relational network into informative formats through built in algorithms and is also capable of dynamic visualization, adapting and changing aspects of the visualization overtime as defined dynamic attributes are changed.

Within the scope of this visualization system there are two dynamic attributes: the period of existence and the score of a node. The period of existence of a node corresponds to

the time when that node's comment is posted in the discussion, the original content submission will always be the first node, and parent comments will always exist before their child comments. The score of a node is used to dynamically affect the size of the node over time, the increase and decrease in size corresponds to the popularity of the comment. The goal of this dual dynamicity is to present to the end user, in a natural and intuitive manner, the growth of the size and popularity of a discussion in relation to the number and popularity of the comments within that discussion.

Implementation

The solution to the lack of data regarding time based changes to the score of content and comment submissions on Reddit is to gather data from Reddit in real time. There are several methods to gather data from Reddit, such as parsing Reddit pages as JSON files, but the most formalized one is to use the Reddit application program interface (API). The Reddit API was designed to allow the automation of a variety of complex functions related to using Reddit and moderation of Subreddits. Subreddit moderators can use the Reddit API to write scripts that automate certain aspects of the moderation process, such as complain handling or automatic keyword detection. Hobbyist users have used the Reddit API to create simple automated applications dedicated to specific functions, such as the user account MetricConversionBot, which automatically detects imperial units within any comment and converts them to metric units in a child comment. Similarly, the user account JiffyBot converts Youtube videos to animated Gif files and rehosts them on an easily accessible website and posts a link to that image in a child comment.

As a central function, the Reddit API allows automated applications to gather and sort through a large amount of data from Reddit. However, there are certain rules and restrictions to using the Reddit API such as a limit of one connection per two seconds from any unique IP address, these limitations were created in order to regulate automated applications and prevent intentional or unintentional overloading of the Reddit servers. To simplify usage of the Reddit API, some independent programmers created the Python Reddit API Wrapper. PRAW is a package that is available to the programming language Python that allows programmers to streamline their development process by handling the data retrieval and Reddit API interface process, automatically bringing the application in line with Reddit API regulations. PRAW also assist programmers by providing better organization to both the process, and the data retrieved through the Reddit API.

Using the Reddit API and PRAW, submissions and comments are retrieved from Reddit as objects with associated variables containing all the relevant information for that submission or comment. Each submission or comment is uniquely identified by a 6 digit alphanumerical string, and the string can be used to directly retrieve the referenced submission or comment page on the Reddit website.

PostgreSQL is an enterprise scale open source relational database system, being both feature-rich and scalable. PostgreSQL was chosen for this project in part due to its ability to handle a large amount of data and in part due to its popularity and open source nature encouraging developers to create additional systems and modules that can interface with it. PSYCOPG2 is used in this project to connect PostgreSQL and Python.

The Real Time Conversation application is written in Python 3.4.3. Using the PRAW and PSYCOPG2 modules, the application moves data from the Reddit API into PostgreSQL by using the available variables of the pre-defined Submission and Comment objects in PRAW.

To start, a couple of sets are made in order to keep track of the submission parsed over the period that the program runs, one set (A) is of all of the unique IDs of submissions that appear on the front page over the 12 hour period, the other set (B) contains the unique IDs of all of the submissions that are being parsed during the same time.

To start parsing data, a list of the Submission from the rising page is retrieved from Reddit using PRAW. For each Submission in the list, if its unique ID is not already in set (B), the unique ID is retrieved and used to create two unique tables in PostgreSQL, one for nodes, storing the comments and one for edges, storing the relationship between comments. After the appropriate tables have been created, the application retrieves each of the submissions in set (B) from PRAW and all of the comments for that submission are placed into a set and recursively parsed to extract the unique ID and current time variable, these variables are used to create a new record in the database, containing additional information parsed from variables such as the body text of the comment. Once the process of parsing all comments is completed, the application returns to the beginning and retrieves the new list of submissions on the rising page, starting the process over.

While this process seems to retrieve redundant data many times, such as submission and comment ID, these are used in conjunction with the current time to uniquely identify each record, variables that very rarely change, such as the body text of the comments, are only re-

recorded if there was actually a change. This is the simplest of several methods that could be used to record information of comment scores at different times; it is also the one that records the most complete information. Alternative methods have been implemented for testing, but the basic concept of checking each individual comment for change in score is still necessary, this method simply records all the data and leaves any organizational process to another portion of the code that does not have to interface with the Reddit API and retrieve information from Reddit.

After 12 hours have elapsed, any discussion that did not reach the front page is dropped in order to reduce the amount of unnecessary information stored in the database. This is achieved by using any unique ID that does not exist in both set (A) and set (B) to retrieve and drop its respective tables. After the data reduction process, the remaining tables for this 12 hour cycle is run through a conversation script that retrieves the necessary data from the tables and forms it into a Gexf file that can be visualized using Gephi. The conversation script uses PSYCOPG2 and the unique ID of each table to query the available data from each database table, organizes the data by nodes and sorts each node's worth of data by time. The conversion script then wraps all of the data with the appropriate Gexf format, adds the header and footer information and writes the entire file out into a user defined directory.

Gexf, or Graph Exchange XML format, is a language that details the data and dynamics of complex network structures. Created by members of the Gephi project solely with the goal of representing networks graphs, the Gexf graph format was created to be able to represent nodes, edges and data associated with a network graph. With the additional ability to handle

any hierarchical structures or dynamic functions that maybe present within a network graph, Gexf is a powerful, extensible and open format that is mature enough for specific applications.

Gephi is an open source network graph visualization platform. Built as a tool to assist analysis of graph data, Gephi enables users to manipulate the structure, shape and color of the visualized graph data in order to assist with data analytics. Gephi is an excellent information visualization system, containing all of the components necessary to an information visualization system, with additional functionality for exploration of other aspects of data analytics and statistics. Due to its open source nature, Gephi allows users to extent its functionality through the usage of plugins, making it a very versatile platform for visualization. However, in the same sense that Photoshop is not the best tool to use to view pictures, Gephi is not suited to casually viewing data visualizations. Gephi is more specialized towards an analytical role with its complex functions.

Because of the community friendly nature of the open source format, Gephi streamlines the installation of plugins. The role of plugins within Gephi is to extent Gephi's information visualization or data analytic functionality. Being open source assists Gephi in this regard and there is an extensive library of plugins available for users seeking specialized functionalities. Plugins used in the Real Time Conversation project were specialized to extent the extraction functionality of Gephi as an information visualization system. Gephi, by default, is able to export visualizations in a couple of different formats including SVG, PDF and PNG files. The plugins Seadragon Web Export and Google Maps Exporter extended the extraction functionality by providing a different method of extracting and displaying visualization data.

Seadragon Web Export allows the exported graphic to be viewed using Microsoft's Seadragon Software through a web browser, Seadragon is the powerhouse behind Mircosoft's Sliverlight, Pivot and Photosynth applications and focuses on smooth browsing of graphics of any size. Google Maps Exporter, on the other hand, extracts the visualization created in Gephi for usage with the Google Maps API, allowing end users to view the extracted visualization in a browser in the same intuitive method that they would use google maps. Both of these plugins give the extracted visualization a couple of additional information visualization functionalities such as smooth zooming and detail views instead of a simple static overview. However, the functionalities of Seadragon Web Export and Google Maps Exporter do not meet to the goal of the Real Time Conversation project, which is to visualize the effects of dynamic time-based changes to data that takes place on Reddit. While the Gephi plugin library is large, there was no plugin that could specifically address the requirement for dynamic visualization exporting. However, Gephi is able to export screenshots of the visualization during each stage of the dynamic visualization, utilizing this function it was possible to use extracted screenshots at each interval of the dynamic visualization to create an animated Gif file that shows the progressive growth of the dynamically visualized network over time. While inconvenient and manual labor intensive, it was able to achieve the goal of exporting a dynamic, time based visualization.

Discussion

The usual focus of information visualization with regards to social networking sites have generally been the visualization of relationships and connections between individuals within a social group, the Real Time Conversation project demonstrated the possibility of dynamically

visualizing discussions that occur over time within a community on social networking sites. The visualizations created from the Real Time Conversation project application met the baseline goals of detailing the growth of both the size and the popularity of discussions on Reddit through a dynamic time-based visualization of comments and their score. When compared to existing static visualizations, dynamic visualization provided additional information regarding the growth of the discussion and retained a similar sense of the discussion structure.

While Reddit is uniquely formatted to encourage community discussions, the same process of gathering conversational data for visualization using a relational network graph is applicable to other social networking sites, as there are usually systematic manners in which users of social networking sites engage in conversations. At the same time, there are several limitations of gathering conversational data that are specific to Reddit. Since Reddit hides certain comments behind hyperlinks to reduce the need to transfer unnecessary data, a certain amount of data is passed over by the data gathering process; attempts to fill in the gaps required excessive connections to Reddit in order to retrieve the additional data and negatively impacted the efficiency of the data gathering process.

Using Gephi as a visualization system allowed end users to engage with the visualization through the usage of the components of a visualization system; however a certain amount of knowledge regarding the use of Gephi was necessary to use this visualization system to its full extent. Exporting visualizations from Gephi caused the visualization to lose several interactive components in addition to requiring a significant amount of manual labor.

While the focus of this project did not include social network analysis or data statistics, visual analysis of a small sample of ten visualizations did show similarities with regards to the growth of the relational network graph and the spread of popularity among the different nodes. From the static visualizations, it can be seen that the discussions are structured to have between one and five significantly sized branches of conversation, of which, one is significantly larger than the others and may contain a significant branch of its own. Of the nodes in the discussion, the submission itself has the most aggregate score, and the first node of the largest or second largest branch will have the second highest aggregate score. When the dynamic visualization is viewed, it is revealed that the submission itself gains most of its score early on in the discussion's lifetime, but the longer the discussion lasts, the most likely that the submissions actually begin losing score towards the end of the discussion's lifetime. The comments start gaining score after the submission has started, however the slowing of the rate at which the submission gains score does not necessarily affect the comment's aggregate score.

This basic analysis seems to suggest that there may be a systematic trend or underlying method to the progression of discussions that take place within communities on Reddit. Due to medium limitations, only the end-state static visualizations of sample set have been included in Appendix B. The result of using this visualization to intuitively recognize patterns in the data regarding discussions on Reddit are encouraging, and indicates that there may be an application for the Real Time Conversation project as a tool to assist social scientists conduct social network analysis or find specific patterns of interest.

Conclusion

In this project I defined a need for data visualization and analysis regarding time based changes to the popularity of online content within the scope of the online social networking site Reddit. With no data readily available for analysis, an application was developed that was capable of logging and exporting time-based changes to content popularity on Reddit for visualization through the information visualization and analysis program Gephi. The visualizations generated by Gephi could be used to analyze the growth of the size and popularity of the discussions over time. While it is outside the scope of this project, this project may potentially be applicable to the fields of social psychology and other related social sciences.

On the basis this project, some additional projects could be pursued. A plugin for Gephi that specializes in the extraction and exporting of dynamic visualization in animated image and video formats could be beneficial to the graph visualization community in the long run. Additionally, future work should be done to extend the gathering and visualization of conversational data generated by Reddit in order to allow the visualization to be viewed in a real time, interactive web format that may serve as a companion or alternative to the standard Reddit web user interface. An extension of the data gathering application that does not restrain the data logging to specific cycles and can continuously gather data may also assist in providing more complete and additional information for future analysis.

References

Boyd, d. m. and Ellison, N. B. (2007), Social Network Sites: Definition, History, and Scholarship. Journal of Computer-Mediated Communication, 13: 210–230. doi: 10.1111/j.1083-6101.2007.00393.x (http://onlinelibrary.wiley.com/doi/10.1111/j.1083-6101.2007.00393.x/full)

Brandes, U., & Wagner, D. (2004). Analysis and Visualization of Social Networks. Graph Drawing Software Mathematics and Visualization, 321-340. doi:10.1007/978-3-642-18638-7_15

Christensen, M., Jansson, A., & Christensen, C. (2011). Online territories: Globalization, mediated practice, and social space. New York: Peter Lang.

Ellison, N. B., Steinfield, C. and Lampe, C. (2007), The Benefits of Facebook "Friends:" Social Capital and College Students' Use of Online Social Network Sites. Journal of Computer-Mediated Communication, 12: 1143–1168. doi: 10.1111/j.1083-6101.2007.00367.x (http://onlinelibrary.wiley.com/doi/10.1111/j.1083-6101.2007.00367.x/full)

Garton, L., Haythornthwaite, C. and Wellman, B. (1997), Studying Online Social Networks. Journal of Computer-Mediated Communication, 3: 0. doi: 10.1111/j.1083-6101.1997.tb00062.x (http://onlinelibrary.wiley.com/doi/10.1111/j.1083-6101.1997.tb00062.x/abstract;jsessionid=0ED35DFA463EF5F92197376433B41FC6.f03t02)

Kashcha, A. (2012, July 13). Visualizing communities of reddit.com. Retrieved April 14, 2016, from http://blog.yasiv.com/2012/07/visualizing-communities-of-redditcom.html

Luse, A., Scheibe, K. P., & Townsend, A. M. (2008). A Component-Based Framework for Visualization of Intrusion Detection Events. Information Security Journal: A Global Perspective, 17(2), 95–107. http://doi.org/10.1080/19393550802039791

Olson, R. S. (n.d.). Redditviz | reddit interest network. Retrieved April 15, 2016, from http://rhiever.github.io/redditviz/#tryptonaut

Olson, R. S., & Neal, Z. P. (2015). Navigating the massive world of reddit: Using backbone networks to map user interests in social media. PeerJ Computer Science, 1. Retrieved April 14, 2016, from https://peerj.com/articles/cs-4/.

Quade, L. (2013, September 13). I made a map of Reddit. (xpost from r/pics) [1787x2157] • /r/MapPorn. Retrieved April 14, 2016, from https://www.reddit.com/r/MapPorn/comments/o9j6a/i_made_a_map_of_reddit_xpost_from_rpics_1787x2157/

Reddit conversation network. (n.d.). Retrieved April 15, 2016, from http://whichlight.github.io/reddit-network-vis/

Scott, J. (2012). SOCIAL NETWORK ANALYSIS (3rd ed.). SAGE. (https://books.google.com/books?hl=en&lr=&id=MJoIGBfYDGEC&oi=fnd&pg=PP2&dq=social+network+&ots=zwDsZ2Zh9f&sig=DD7eEQXeBhi36BxLszR_wtR7Uag#v=onepage&q=social%20network&f=false)

Serrat, O. (2010). Social Network Analysis. Asian Development Bank, Washington. doi:10.1007/978-3-531-19340-3 (http://digitalcommons.ilr.cornell.edu/intl/206/)

Shneiderman, B and Plaisant, C. (2005). Designing the User Interface (4th ed.). Boston: Pearson Education, Inc.

Virdee, K. (2014, January 1). Visualizing discussions on Reddit with a D3 network and Embedly. Retrieved April 14, 2016, from http://blog.embed.ly/post/57097477000/visualizing-discussions-on-reddit-with-a-d3

Wasserman, S., & Faust, K. (1994). Social network analysis: Methods and applications. Cambridge: Cambridge University Press.

Appendix A

Python 3.4.3 source code for the RTC application

```
import praw, psycopg2, time, sys, os, urllib.error
from pprint import pprint


def Reddit_crawler(Hours_to_run,DBConnection):
    pprint('running Reddit crawler')
    sleeptime = 60
    Reddit_object = praw.Reddit("real time converstation scraper")
    Watching_submissions = set()
    Cached_submissions = set()
    Cached_comments = set()
    Front_page_set = set()
    Hour_count = time.time() + 3600*Hours_to_run
    Error_count = 0
    Time_not_up = True
    while Time_not_up:
        try:
            Front_page = Reddit_object.get_front_page()
            for sub in Front_page:
                if sub.id not in Front_page_set:
                    Front_page_set.add(sub.id)

            Rising_page = Reddit_object.get_rising()
            for sub in Rising_page:
                if sub not in Watching_submissions:
                    Watching_submissions.add(sub)

            for sub in Watching_submissions:
                if sub not in Rising_page:
                    submission = Reddit_object.get_submission(submission_id=sub.id)
                else:
                    submission = sub

                DBCursor = DBConnection.cursor()

                if submission.id not in Cached_submissions:
                    try:
                        author = str(submission.author) + ' -- OP'
                    except AttributeError:
                        author = 'none'
                    DBCursor.execute("CREATE TABLE node{} (id VARCHAR, label VARCHAR, created INT,
score INT, content VARCHAR);".format(submission.id))
                    DBCursor.execute("CREATE TABLE edge{} (source VARCHAR, target VARCHAR, id
VARCHAR, created INT);".format(submission.id))
                    query = "INSERT INTO node{} (id, label, created, score, content) VALUES (%s, %s, %s, %s,
%s);".format(submission.id)
                    data = (submission.name, author, submission.created_utc, submission.score,
submission.selftext + submission.title)
                    DBCursor.execute(query, data)
```

```
            Cached_submissions.add(submission.id)
        else:
            if submission.edited:
                Updated_content = submission.selftext
            else:
                Updated_content = 'N/A'
            query = "INSERT INTO node{} (id, label, created, score, content) VALUES (%s, %s, %s, %s,
%s);".format(submission.id)
            data = (submission.name, author, time.time(), submission.score, Updated_content)
            DBCursor.execute(query, data)
            pass

        Flat_comments = praw.helpers.flatten_tree(submission.comments)

        for comment in Flat_comments:
            try:
                author = str(comment.author)
            except AttributeError:
                author = 'none'
            if isinstance(comment, praw.objects.Comment):
                if comment.edited:
                    Updated_content = comment.body
                else:
                    Updated_content = 'N/A'
                if comment.id in Cached_comments:
                    try:
                        query = "INSERT INTO node{} (id, label, created, score, content) VALUES (%s, %s,
%s, %s, %s);".format(submission.id)
                        data = (comment.name, author, time.time(), comment.score, Updated_content)
                        DBCursor.execute(query, data)
                        pass
                    except UnicodeEncodeError:
                        pprint('UnicodeEncodeError')
                        pass
                else:
                    try:
                        query = "INSERT INTO node{} (id, label, created, score, content) VALUES (%s, %s,
%s, %s, %s);".format(submission.id)
                        data = (comment.name, author, comment.created_utc, comment.score,
Updated_content)
                        DBCursor.execute(query, data)

                        query = "INSERT INTO edge{} (source, target, id, created) VALUES (%s, %s, %s,
%s);".format(submission.id)
                        data = (comment.name, comment.parent_id, comment.id, comment.created_utc)
                        DBCursor.execute(query, data)

                        Cached_comments.add(comment.id)
                    except UnicodeEncodeError:
                        pass
            else:
                if comment.id not in Cached_comments:
                    try:
                        query = "INSERT INTO node{} (id, label, created, score) VALUES (%s, %s, %s,
%s);".format(submission.id)
                        data = (comment.name, 'more comments', time.time(), 1)
```

```
                        DBCursor.execute(query, data)

                        query = "INSERT INTO edge{} (source, target, id, created) VALUES (%s, %s, %s,
%s);".format(submission.id)
                        data = (comment.name, comment.parent_id, comment.id, time.time())
                        DBCursor.execute(query, data)

                        Cached_comments.add(comment.id)
                    except UnicodeEncodeError:
                        pass
                else:
                    pass

            DBConnection.commit()
            DBCursor.close()
        pprint('time left: ' + str(Hour_count - time.time()))
    except urllib.error.HTTPError as e:
        if e.code in [429, 500, 502, 503, 504]:
            pprint ("Reddit is down (error {}), sleepin for {} seconds".format(e.code, sleeptime))
            time.sleep(sleeptime)
            pass
        else:
            raise
    except:
        pprint ("Unexpected Error")
        if Error_count < 15:
            Error_count += 1
            time.sleep(sleeptime)
            pass
        else:
            raise

    if time.time() > Hour_count:
        Time_not_up = False

if not Time_not_up:
    pprint("time's up")
    Drop_set = set()
    DBCursor = DBConnection.cursor()
    Pass_set = set()
    Pass_set = Cached_submissions.intersection(Front_page_set)
    try:
        Drop_set = Cached_submissions - Front_page_set
        for name in Drop_set:
            DBCursor.execute("DROP TABLE node{}".format(name))
            DBCursor.execute("DROP TABLE edge{}".format(name))
            pprint('Dropped {}'.format(name))
        DBConnection.commit()
        DBCursor.close()
    except urllib.error.HTTPError as e:
        if e.code in [429, 500, 502, 503, 504]:
            pprint ("Reddit is down (error {}), sleeping for {} seconds".format(e.code, sleeptime))
            time.sleep(sleeptime)
            pass
        else:
            raise
```

```
        except:
            pprint ("Unexpected Error")
            if Error_count < 15:
                Error_count += 1
                time.sleep(sleeptime)
                pass
            else:
                raise
        pprint("one done, resetting")
        Cached_submissions.clear()
        Watching_submissions.clear()
        Cached_comments.clear()
        Front_page_set.clear()
        return Pass_set


def Gexf_maker(Table_name,Directory,DBConnection):
    DBCursor = DBConnection.cursor()

    Dup_comments = set()
    Node_set = set()
    Edge_set = set()
    Comment_set = set()

    pprint('getting {}'.format(Table_name))
    DBCursor.execute("SELECT * FROM node{};".format(Table_name))
    Comment_set = DBCursor.fetchall()
    DBCursor.execute("SELECT * FROM edge{};".format(Table_name))
    Edge_set = DBCursor.fetchall()
    DBCursor.close()

    Attvalue_data = {}
    Creation_data = {}
    End_time = 0
    Start_time = 9999999999


    for row in Comment_set:
        if row[0] not in Dup_comments:
            Dup_comments.add(row[0])
            Node_set.add(row)

            Row_list = []
            Row_list.append(row)
            Dict_entry = Row_list

            Dict_key = row[0]

            Creation_entry = row[2]
            Attvalue_data[Dict_key] = Dict_entry
            Creation_data[Dict_key] = Creation_entry
            if row[2] > End_time:
                End_time = row[2]
            if row[2] < Start_time:
                Start_time = row[2]
        else:
```

```
            Dict_key = row[0]
            Original_time = Creation_data[Dict_key]
            Original_list = list(Attvalue_data[Dict_key])
            Original_list.append(row)

            Dict_entry = Original_list
            Attvalue_data[Dict_key] = Dict_entry
            if row[2] < Original_time:
                Creation_data[Dict_key] = row[2]
            if row[2] > End_time:
                End_time = row[2]
            if row[2] < Start_time:
                Start_time = row[2]

    Node_data = ''

    for node in Node_set:
        node_list = list(Attvalue_data[node[0]])
        node_list.sort(key=lambda tup: tup[2])
        Node_att_string = ''
        for index in range(len(node_list)):
            if index+1 < len(node_list):
                if node[1] == 'more comments':
                    New_string = '<attvalue for="score" value="{}" start="{}"
end="{}"></attvalue>'.format(1,node_list[index][2]-Start_time,node_list[index+1][2]-1-Start_time)
                    Node_att_string += New_string#
                else:
                    New_string = '<attvalue for="score" value="{}" start="{}"
end="{}"></attvalue>'.format(node_list[index][3],node_list[index][2]-Start_time,node_list[index+1][2]-1-
Start_time)
                    Node_att_string += New_string
            else:
                if node[1] == 'more comments':
                    New_string = '<attvalue for="score" value="{}" start="{}"
end="{}"></attvalue>'.format(1,node_list[index][2]-Start_time,End_time+1-Start_time)
                    Node_att_string += New_string
                else:
                    New_string = '<attvalue for="score" value="{}" start="{}"
end="{}"></attvalue>'.format(node_list[index][3],node_list[index][2]-Start_time,End_time+1-Start_time)
                    Node_att_string += New_string
        Node_string = '<node id ="{}" label="{}" start="{}"><attvalues>{}</attvalues></node>'.format(node[0],
node[1], Creation_data[node[0]]-Start_time, Node_att_string)
        Node_data += Node_string


    Edge_data = ''

    for edge in Edge_set:
        Edge_string = '<edge source="{}" target="{}" start="{}"></edge>'.format(edge[0], edge[1], edge[3]-
Start_time)
        Edge_data += Edge_string

    Edges = '<edges>{}</edges>'.format(Edge_data)
    Nodes = '<nodes>{}</nodes>'.format(Node_data)
    Attribute = '<attribute id="score" title="score" type="float"></attribute>'
```

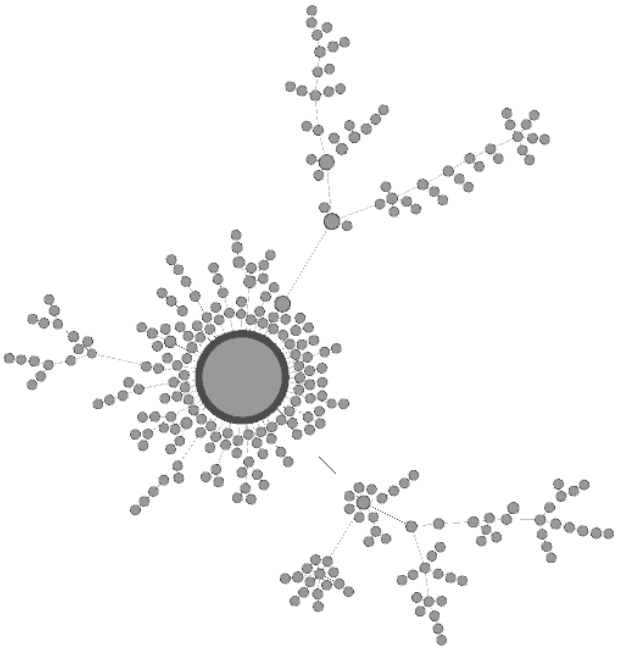
```
    Attributes = '<attributes class="node" mode="dynamic">{}</attributes>'.format(Attribute)
    Graph = '<graph defaultedgetype="undirected" timeformat="double" timerepresentation="interval"
mode="dynamic">{}{}{}</graph>'.format(Attributes, Nodes, Edges)
        Gexf = '<gexf xmlns="http://www.gexf.net/1.3" version="1.3" xmlns:viz="http://www.gexf.net/1.3/viz"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.gexf.net/1.3
http://www.gexf.net/1.3/gexf.xsd"><meta><creator>Gephi
0.9.1</creator><description></description></meta>{}</gexf>'.format(Graph)
    Header = '<?xml version="1.0" encoding="UTF-8"?>'

    os.chdir(Directory)
    pprint('writing ' +Table_name+ ' to ' +Directory)
    w = open(Table_name+'.gexf', 'w')
    print (Header+Gexf, file=w)
    w.close()

Rerun = True
while Rerun :
    Hours_to_run = int(input('Enter number of hours to run for(default is 12): ') or '12')
    Database_name = input('Enter database name(default is RTC): ') or 'RTC'
    Directory = input('Enter Gexf output directory(default is C:\Programming\Gexf): ') or
'C:\Programming\Gexf'
    DBConnection = psycopg2.connect("dbname={} user=postgres
password=root".format(Database_name))
    Data = Reddit_crawler(Hours_to_run,DBConnection)
    for table in Data:
        Gexf_maker(table,Directory,DBConnection)
    Rerun_check = input('Run again?(default is yes)[Y/N]: ') or Y
    Rerun = False
    if Rerun_check.lower() in ['y', 'yes']:
        Rerun = True
        pprint('Rerunning')
    else:
        pprint('Done')
        sys.exit()
if not Rerun:
    pprint('something went wrong, exiting.')
    sys.exit()
```

Appendix B

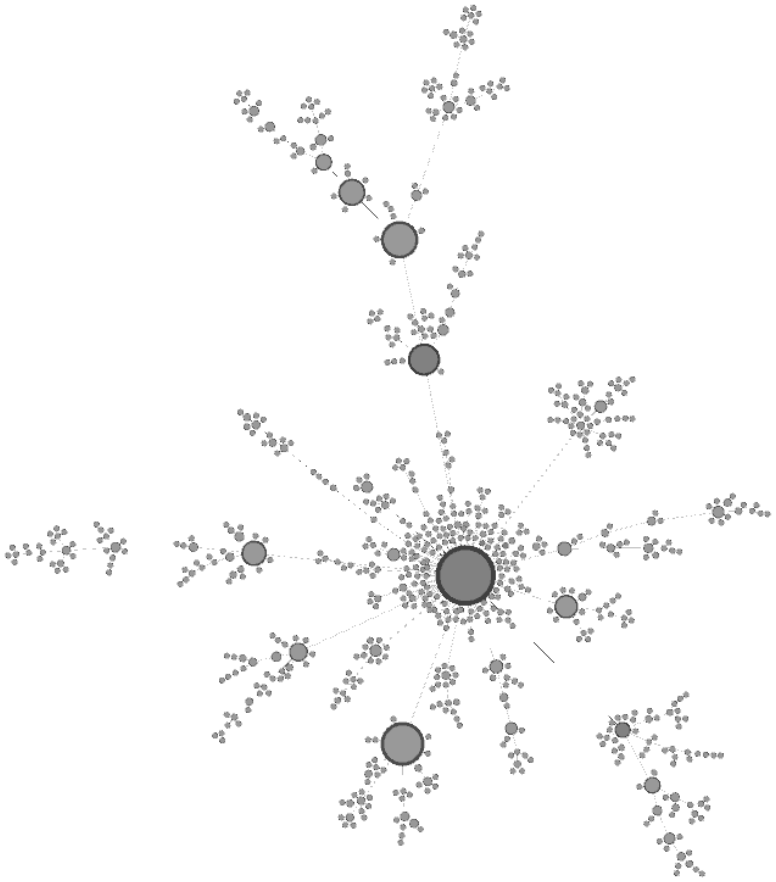
Samples of static visualization outputs



Submission ID: 4gbl5k

Title: “How Las Vegas, BLVD outsmarts vandalism.”

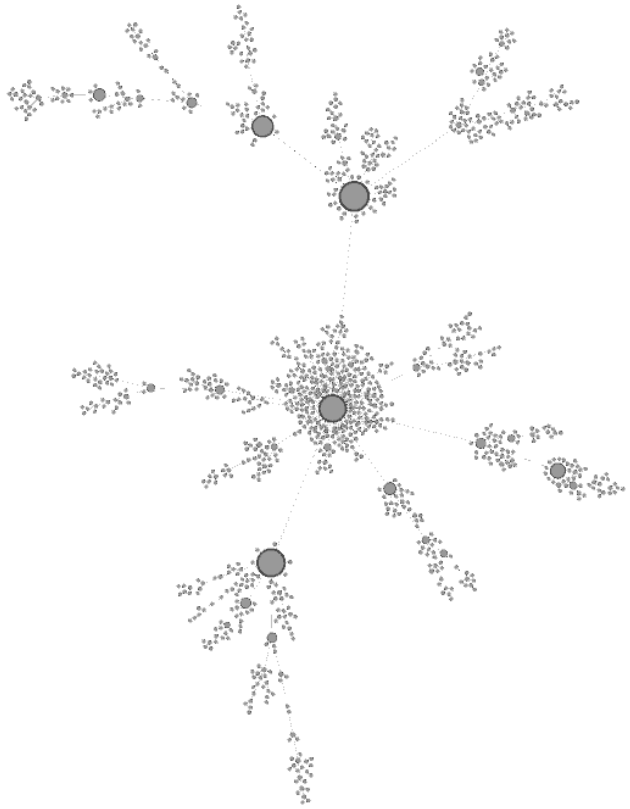
Subreddit: R/mildlyinteresting



Submission ID: 4gk3am

Title: “The Rock posts first still from "Baywatch"”

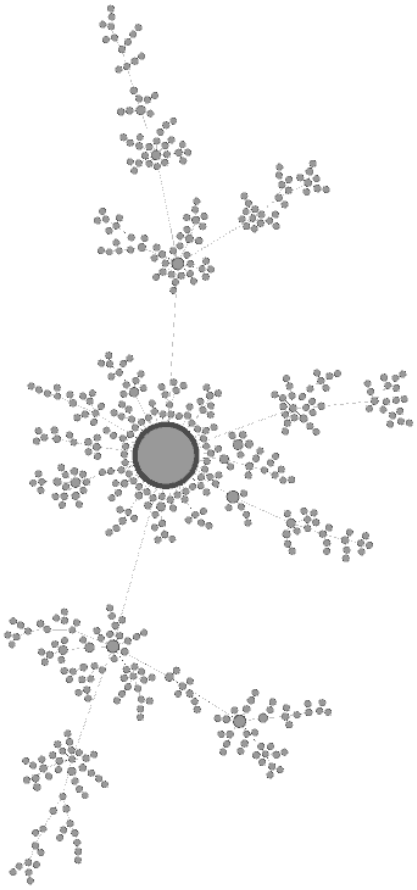
Subreddit: R/movies



Submission ID: 4gkgsg

Title: “TIL Mother Teresa considered suffering a gift from God and was criticized for her clinics' lack of care and malnutrition of patients.”

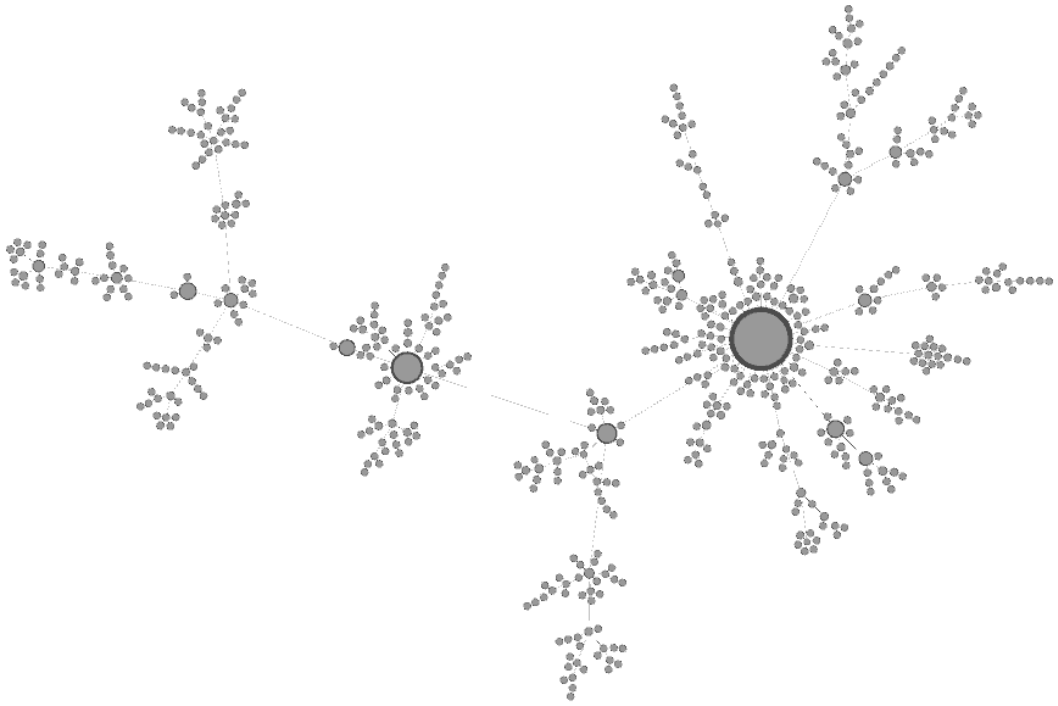
Subreddit: R/todayilearned



Submission ID: 4glsld

Title: “ELI5:What's the 'point of no return' for regenerating muscles and bones? If you loose a finger, that's it, but if you get a really nasty cut, most will heal.”

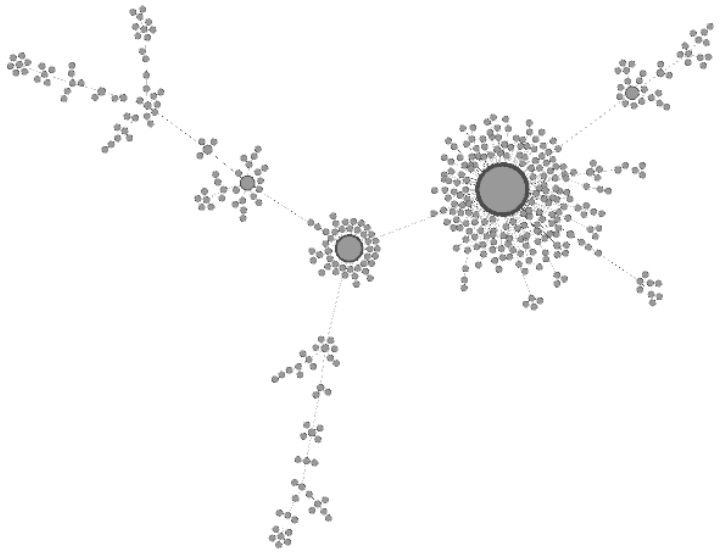
Subreddit: R/explainlikeimfive



Submission ID: 4gltdl

Title: “KATE WHAT THE FUCK?”

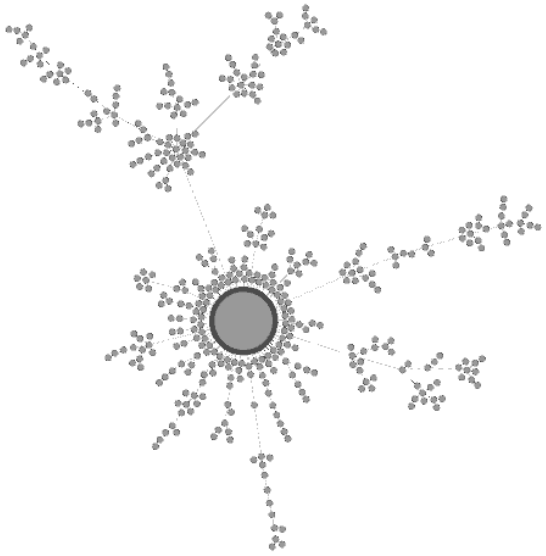
Subreddit: R/videos



Submission ID: 4gm3c3

Title: “Not all heroes wear capes”

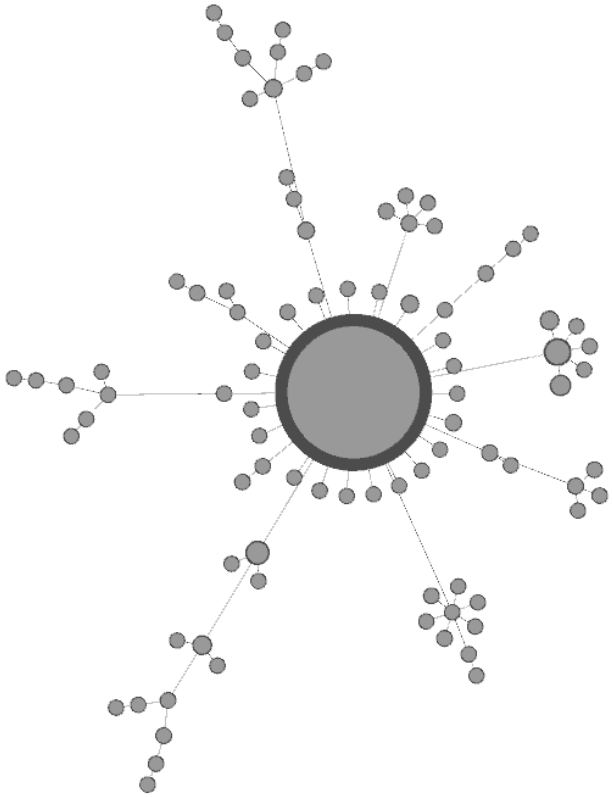
Subreddit: R/funny



Submission ID: 4gm06s

Title: “LPT - To all cops with a '7-11' in their district: stop by after midnight and you can get food for the homeless”

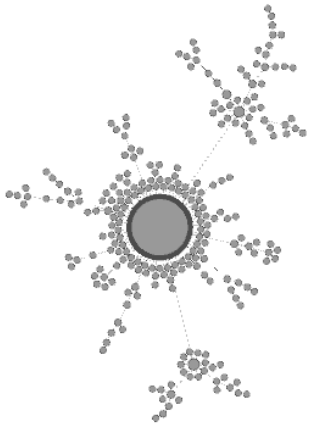
Subreddit: R/LifeProTips



Submission ID: 4gm8t9

Title: “Tornado Probabilities by Day [OC]”

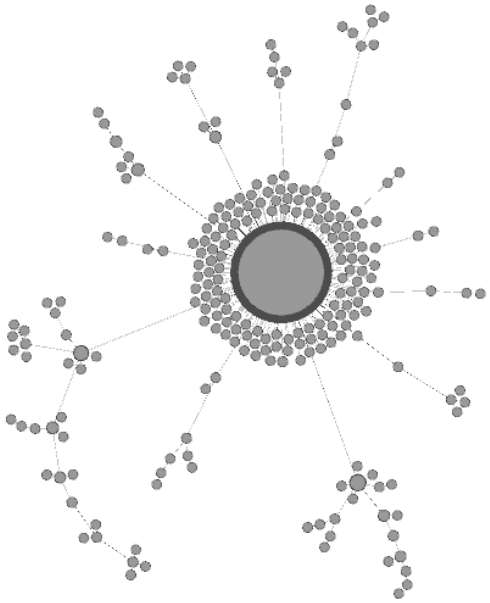
Subreddit: R/dataisbeautiful



Submission ID: 4gmaeb

Title: “[Image] The Moon”

Subreddit: R/GetMotivated



Submission ID: 4gl6gm

Title: “I accidentally painted a snow covered forest with my router shavings.”

Subreddit: R/mildlyinteresting