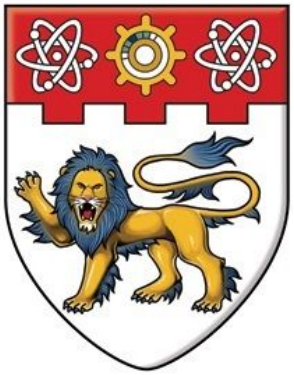


NANYANG TECHNOLOGICAL UNIVERSITY

KANDINSKY:

Visualization of Social Comments on Mobile Devices

By

Castillo Clarence Fitzgerald Gumtang

School of Computer Science and Engineering

2019/2020

NANYANG TECHNOLOGICAL UNIVERSITY

Project ID: SCSE19-0132

KANDINSKY:

Visualization of Social Comments on Mobile Devices

Submitted by: Castillo Clarence Fitzgerald Gumtang

Matriculation Number: U1622291F

Supervisor: A/P Sourav Saha Bhowmick

School of Computer Science and Engineering

Submitted in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Computer Science of the Nanyang Technological University

2020

Abstract

Used by half of the world's population, social media allows users to create and share content in various formats which can collectively be termed as anchor posts. Such anchor posts may attract hundreds of comments from many social users, initiating online conversations, which could generate meaningful insights for both individuals and organizations. Given the large volume of conversations happening, generating insights by simply browsing the comments is tedious, time-consuming and ineffective. Thus, there is a need for a user-friendly and effective tool to facilitate the in-depth analysis and understanding of the social discussions landscape of an anchor post. Moreover, given that mobile devices are one of the fastest growing platforms with which people access social media, it is important to realize the said visualizations on mobile devices.

This project has developed KANDINSKY Mobile, a mobile-based end-to-end visualization application, for supporting the user on analysing copious and complicated social conversations. The challenges are largely due to the unique characteristics of mobile devices, including the small screen size, limiting computing resources, network constraint, and different ways users navigating mobile interfaces.

Inspired by abstract paintings of renowned artist Wassily Kandinsky, KANDINSKY Mobile's visualization scheme is based on a collection of colorful circles and concentric circles. It allows mobile users to visualize a bird's-eye view of the conversation landscape, analyze evolution of discussions over time, perform keyword search on the discussion threads, and find comments that are topically similar to a comment-of-interest.

Based on usability evaluation, the visualization design and human computer interaction design of KANDINSKY Mobile are both intuitive and satisfying for users. There are several directions of future work for KANDINSKY Mobile, including support for more social media platforms and languages, and incorporation of sentiment analysis and function.

Acknowledgements

I would like to express my sincerest thanks and gratitude to my project supervisor, Dr. Sourav Saha Bhowmick, for guiding me patiently throughout the project. The guidance and direction provided was greatly helpful in understanding and developing the project.

To my fiance, Tu Anqi, a fellow NTU BCG classmate, who has extended her moral and technical support throughout my entire time in NTU and especially during the challenging trials faced in this FYP project.

Table of Contents

Acronyms

Abbreviation	Definition
API	Application Program Interface
CSS	Cascading Style Sheets
HTML5	Hypertext Markup Language
IDE	Integrated Development Environment
CRUD	Create, read, update and delete
LDA	Latent Dirichlet Allocation
UI	User Interface
UX	User Experience

List of Figures

List of Tables

1. Introduction

This chapter provides the background information on social conversations happening on social media sites and the motivation for developing a mobile-based system to visualize such conversations, followed by a description of the project scope, major challenges, and the report organization.

1.1 Motivation

Social media has become an integral part of the lives of people from all backgrounds today. With more than 3.5 billion users worldwide in 2019, social media has roughly 46 percent of the world's total population (DataReportal, 2019). It allows users to create and share content in the form of texts, images, or videos, which are collectively termed as anchor posts (Figure 1). Such anchor posts may attract comments from many commentators, initiating online conversations. The social discussions could generate meaningful insights for both individuals and organizations. For example, individual buyers can make more informed purchase decisions by checking conversations about a certain brand's anchor posts. Non-profit organizations can gain insight on public opinions on their campaigns from discussions associated with their anchor posts.

Given the large volume of conversations happening, generating insights by simply browsing the comments is tedious, time-consuming and ineffective. Even though social media sites provide some analytical tools, the majority of them are very primitive. For example, Youtube allows the user to sort comments related to an anchor post by popularity or time. With its limited features, such tools fail to support analysis and understanding of the social discussions landscape associated with an anchor post. Thus, there is a need for a more user-friendly and effective tool to facilitate such a process.

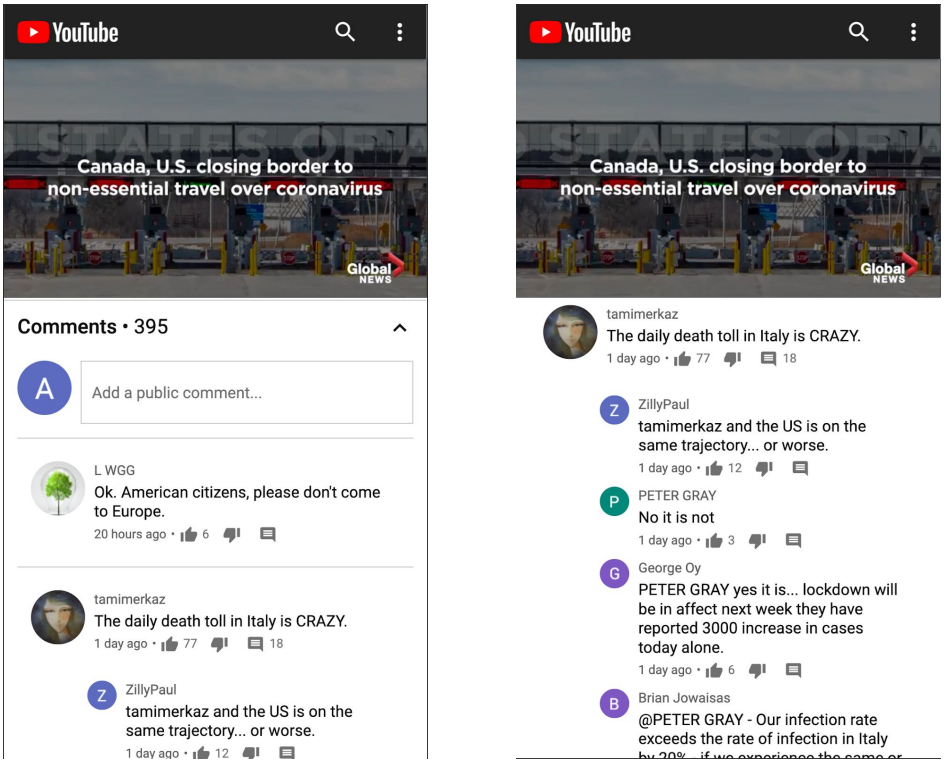


Figure 1: Social Discussions Associated with Anchor Posts on YouTube

An end-to-end visualization system called KANDINSKY (Figure 2a) has been developed for desktop machines to support multi-faceted visualization of social discussions associated with an anchor post. Being inspired by abstract art, the visualization scheme is based on a collection of colorful circles and concentric circles, which are based on the famous abstract arts entitled "Squares with Concentric Circles" and "Several Circles" (Figure 2b) by Russian painter and art theorist Wassily Kandinsky (1866-1944) (Lui, Bhowmick, and Jatowt, 2019).

Given that mobile devices are the most popular ways for people to access social media platforms (Figure 3), it is important to realize such visualization on such devices.

More than just mobile platform support, KANDINSKY Mobile extends and differs from its desktop counterpart in several ways. Aside from minor improvements and several bug fixes, the mobile application features an overhauled reimplementation of the visualisation component made to fit mobile device screens and a new spectrum

mode to filter new comments made within a specified time range extending the old method of comparing two different points in time.

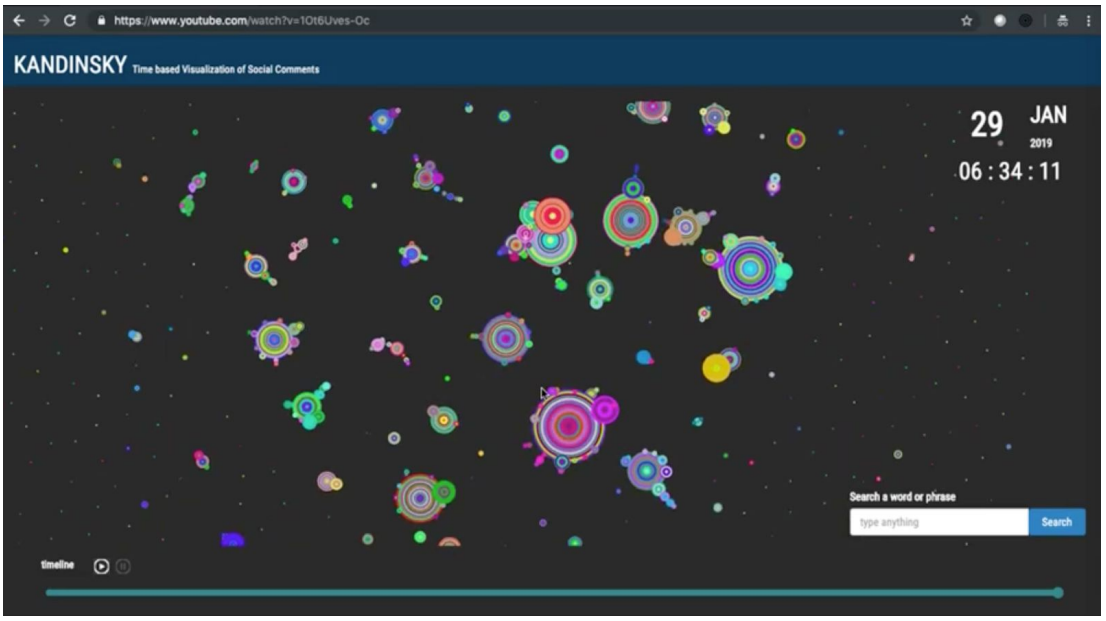


Figure 2(a): KANDINSKY for Desktop Machines

Figure 2(b): Squares with Concentric Circles (1913) and Several Circles (1926)

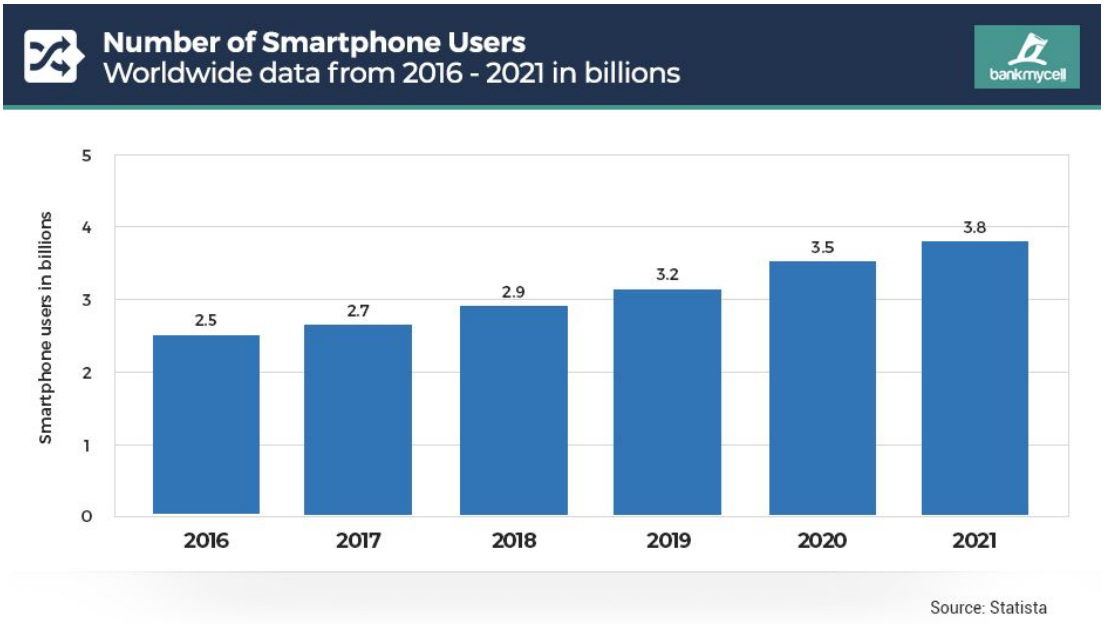


Figure 3: Number of Smartphone Users

1.2 Purpose and Scope

The purpose of this project is to develop KANDINSKY Mobile, which extends KANDINSKY to mobile devices and allows the mobile users to visualize a bird's-eye view of the conversation landscape, highlighting discussion threads that have garnered considerable attention and interactions between various comments and users. It should also provide functions including visualization of evolution of discussions over time, keyword search on the discussion threads, and visualizing comments that are topically similar in content to a comment-of-interest.

1.3 Challenges

Due to the unique characteristics of mobile devices, several challenges are involved in developing such a visualization system on mobile devices, including the small screen size, touchscreen interaction, limited single-touch interaction, limited computing resources, and network restrictions.

Owing to the need for portability, mobile devices generally have small-sized screens. However, the limited screen size makes it difficult to efficiently present information and help users navigate to and from the information they want (Huang, 2002). Given the potentially voluminous comments associated with an anchor post, it is difficult to squeeze all information in a small display space. According to Wang and Sajeev (2007), to display information that is well suited for larger screens, such as a large volume of information, the information has to be segmented into many small presentation units to fit into the small screen of mobile devices, making it difficult to effectively organize information. Besides, With less information shown on the screen, the application will also require the user to increase interaction level to get to desired information (Kamba et al., 1996), giving problems of navigation. For instance, considering searching tasks, Jones and others (2002) have shown that screen size has a major impact on user performance.

Bsdeis, touchscreens have completely changed mobile user interface and interaction design (Punchoojit & Hongwarittorrn, 2017). When users interact with touch screen mobile phones, the habits are drastically different from when they operate with desktops. The different ways that people interact with mobile devices poses an even greater challenge.

The combination of a small size of mobile devices and touchscreen interaction also give rise to the limited single-touch interaction, thus leading to the need for fat-thumb friendliness. Though most mobile devices are equipped with multi-touch screens to support content manipulations by multiple fingers, thumbs remain the most popular input method ("Mobile Usability Testing: Fat Finger Syndrome", 2019). A study done on how people hold and interact with their mobile phones shows that 49% of users manage their device single-handedly, 36% cradle their phones (i.e., using two hands to hold a mobile phone but only one hand to touch the screen or buttons for better stability), and the only 15% are two-handed cases (Hoober & Berkman, 2012). For the majority who use devices single-handedly, performing multi-touch actions on a device with just one hand required awkward hand postures

required, thus making interaction limited to only a single-touch (i.e., the thumb) (Boring et al., 2012). Of the 36% that cradle their device in one hand and interact with the other, 72% use their thumb. For the 15% user using two hands, users are still more adept at tapping the screen with their thumbs. Figure 4 illustrates Common ways people hold and touch mobile phones ("UX for Mobile: The Rise of Fat-Finger Design"), showing that most users rely on single-touch and most use their thumbs as the only input method.

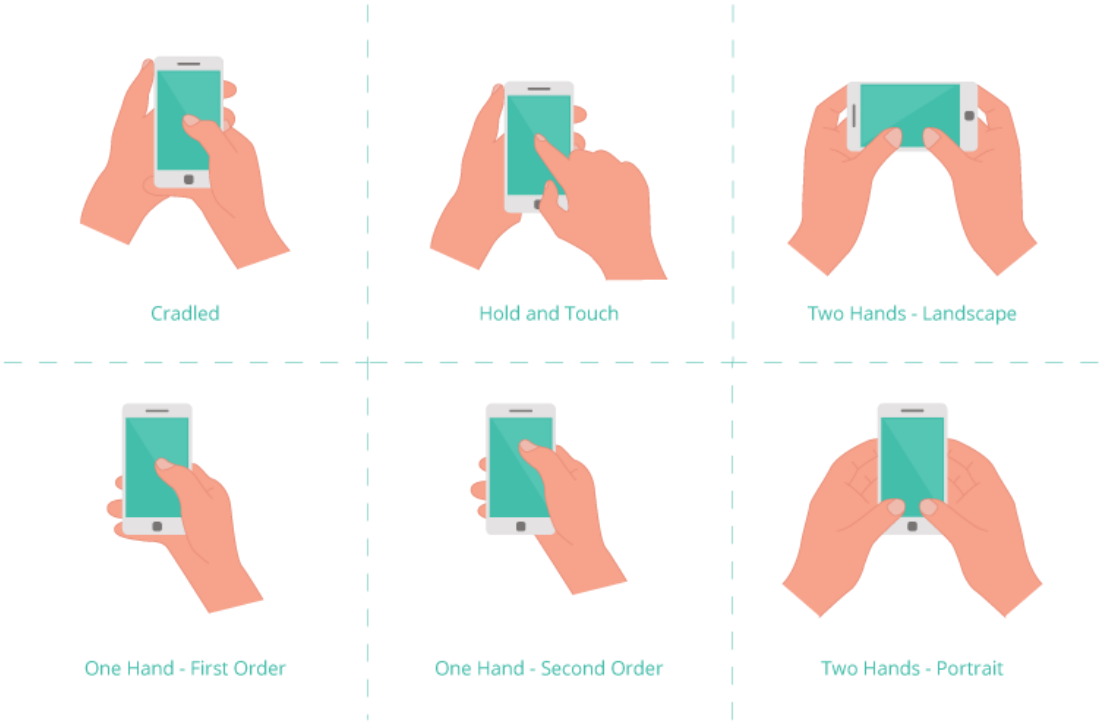


Figure 4. Common ways people hold and touch mobile phones

When the interaction is limited to a finger touch and not a cursor, the size of the touch targets as well as the space between elements need to be carefully designed. The greater the area, the less chance of user touch error. However, area is limited and precious in mobile devices. Thus, it is critical to design mobile applications with the thought that most users are using their thumbs, and no thumb size is equal, in mind.

Also, when the rest fingers are grasping the device, the range of motion available to the thumb is more limited, thus limiting the area of the screen the user's thumb can

reach. This poses another challenge for the layout positioning of touch targets in the screen to ensure accessibility by users.

Furthermore, the design for mobile devices comes with hardware challenges. Due to the limitations in size and weight, mobile devices normally have resource scarcity and thus reduced processing capacity (Fernando et al., 2013). As the analysis process normally requires certain computation operations, the heavy computation might pose a strain on the limiting computing resources available on most mobile devices.

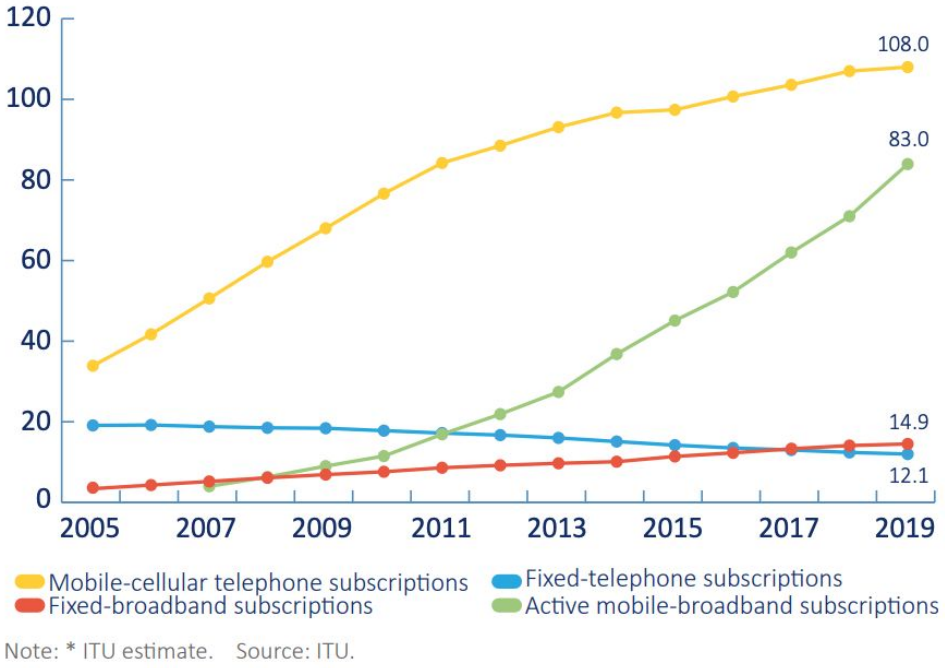


Figure 5: Strong Growth of Mobile-broadband Subscriptions

With the mobile broadband subscriptions continuing to grow strongly worldwide as shown in Figure 5 (ITE Publications, 2019), more people are expected to use the mobile application with cellular data whose billing is always usage-based. When data is limited or expensive, mobile Internet users are extremely conscious about usage costs (Mathur, A., Schlotfeldt, B., & Chetty, M., 2015). The total comment volume of popular anchor posts can be potentially large, posing a challenge in helping users to manage their data to optimize costs.

2. Related Work

This chapter explores past research projects on information visualization, visualization of social discussions, and human computer interaction design for mobile devices in consideration of the aforementioned challenges.

2.1 Information Visualization

Information exploration should be an enjoyable experience, but many applications for information visualization cause anxiety due to poor User Interface (UI) design (Wurman, 1989). Shneiderman (1996) summarizes many visual design guidelines and provides an excellent framework for designing information visualization applications - “Overview first, zoom and filter, details-on-demand, relate, history, and extracts”. This design strategy gives the user more control over the amount and type of information received.

Overview: In the implemented application, the user should be provided a concise zoomed out overview of the social comments landscape to avoid data overload. As human beings have fairly limited cognitive processing capacity, the excessive data fails to provide decision-making values.

Zoom: Since the user typically has an interest in some particular portion of the entire information collection, the application should allow the user to zoom into certain groups of comments. The zooming should be smooth enough to help users preserve a sense of position and context.

Filter: Since dynamic queries applied to the information collection is one of the important ideas in information visualization (Ahlberg et al., 1992), it is important to allow the user to filter out interesting contents. By doing so, users can control the displaying contents, thus easily focusing on their interested contents.

Details-on-demand: The function of allowing the user to dive into the details of an item or group is also important. One applicable approach is to let the user to simply click on a certain circle representing a comment, and then provide its details.

Relate: Viewing relationships among items provides possibility for more in-depth interpretation and analysis. For KANDINSKY, the user can be allowed to view relationships between a certain time on the timeline and the number of comments during that period on the bar chart.

History: As information exploration is inherently a process with many stops, keeping a history of actions to support undo and replay is important.

Extract: Once the user finds desired information, it would be useful to be able to extract the information and save it to a file in a format which would facilitate other uses such as emailing, printing or inserting.

2.2 Visualization of Social Discussions

Researches have been conducted on finding effective ways to visualize social discussions, all of which focus on different analysis angles and adopt alternative visual representations.

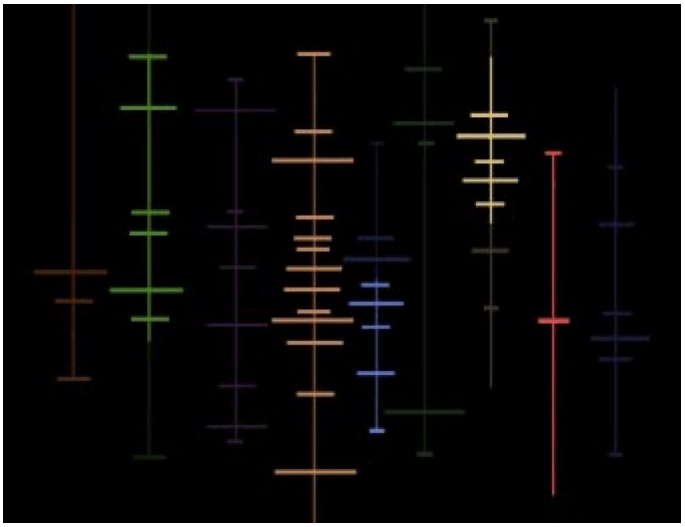


Figure 6: Conversation Landscape for Chat Circles Archives (Donath et al., 1999)

Donath and others (1999) implemented *Chat Circles*, a graphical interface for visualizing synchronous conversations. The interface represents each participation by a colored circle on the screen in which his or her posts appear. The interface *Conversation Landscape* (Figure 6) was also developed to visualize the conversational archive of *Chat Circles*. It provides a 2-dimensional visualization of a chat group where the x-axis represents participants identified by color and the y-axis represents timeline of postings by participants. Each post is shown as a horizontal line, with the length reflecting the message's length. This design provides a bird's-eye view of the activeness of participants in the discussion over time and the interaction patterns of the conversation, making dominant users and surge of discussions recognizable. The problem with this design is that, for discussion among a large number of participants but with low contribution rate, it may render a long horizontal axis but sparse lines.

*Figure 7: Squarified Treemap Layout (*Engdahl *et al., 2005)*

Engdahl and others (2005) proposed an innovative way of visualizing threaded discussion forums on compact displays called *Squarified Treemap Layout*. The technique uses squarified treemaps to display the contents of discussion forums. Each thread from discussion forums is rendered as a colored rectangle in a treemap (Figure

7). The size of the rectangle is either proportional to the number of articles in the thread, or reflects the relevance of the query for that thread when searching is performed. The user is first presented an overview of the top level threads for each forum. As the stylus is moved across the screen, the details of the selected thread is displayed in the popup as shown in Figure X. This approach makes use of 100% of the limited screen space and provides an intuitive overview of the discussion landscape, allowing the user to easily compare popularity among threads. However, the ability to retrieve detailed information and analyze temporal trends is missing.

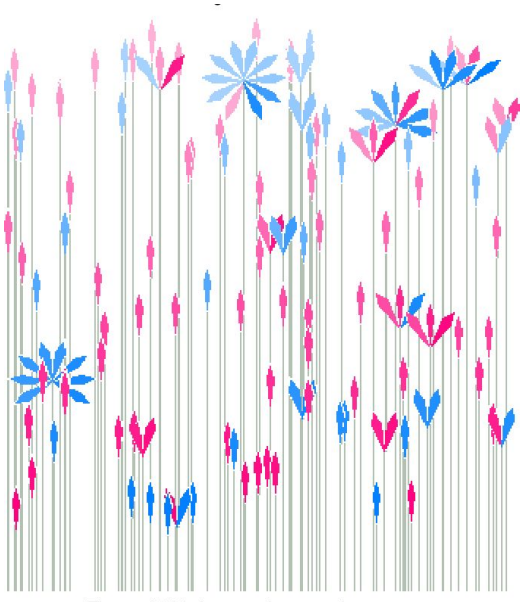


*Figure 8: PeopleGarden (*Xiong and Donath*, 1999)*

Xiong and Donath (1999) designed *PeopleGarden*, which uses data portraits to represent users based on their interactions in a social environment (Figure 8). Each user is portrayed as a flower, whose petals represent the user's posts, and height reflecting the amount of time the user has been in the discussion. Together, users' interactions in the social environment form a garden. This design achieves compact information representation about participants involved in online discussions, but fails to provide details and temporal differences of a discussion. Furthermore, flowers might overlap and become difficult to analyze once the more participants are involved.

System	Advantage	Disadvantages
Conversation Landscape	- Easy to observe the activeness trend - Obvious comparison across each user's activeness	- Ineffective for inactive discussions involving many participants.
Squarified Treemap Layout	- A quick and intuitive overview of the contents such as most discussed threads - Optimized usage of limited screen space	- Loss of details - Unavailable temporal analysis
PeopleGarden	- Compact information of participants activities	- Loss of details - Unavailable temporal analysis - Unsuitable for discussions involving many participants.

Table 1: Summarization of Related Work

2.3 User Interface Design for Mobile Devices

For mobile applications, the need to find effective user interfaces for mobile devices is as critical as solving technological problems and on working out market strategies (Marsden, 2000). Many papers have been published in the field of data visualization on small screens, providing recommendations on our UI design.

In order to save space on small screens, Noirhomme-Fraiture and others (2015) suggested to use concise and precise information. The first implication is to use as few words as possible. However, this does not mean necessarily to use technical words because the ease of understanding of vocabulary should never be compromised. The second implication is to simplify the visualization by using analytical methods to summarize and reduce the amount of information before

information. For example, when displaying the time-series trend of comment columns, a bar chart can be used to summarize the number of comments for each equal-width interval. The third implication is that the visualization should optimise the screen space usage by using maximum of the available space on the screen. This signals the importance of automatically finding the suitable scale of circles given the screen size.

To overcome the navigation problem on small screens, it is recommended to replace tapping with vertical scrolling (Jones et al., 2002). Giller and others (2003) observed that users performed significantly better when they could scroll, instead of tapping on widget elements tabs for page-to-page navigation, and that scrolling vertically rather than horizontally led to better user experience. In the application, vertical scrolling should be used when displaying the long list of comments, such that the number of taps can be minimized (Raneburger et al, 2013).

To overcome the limited interaction with a finger touch (i.e., the thumb), the application must be fat-finger friendly, by carefully designing the size of touch targets. Parhi, Karlson, and Bederson (2006) recommend that target sizes should be at least 9.2 mm for discrete single-target (e.g. selecting a button) tasks and 9.6 mm serial for multi-target (e.g. text entry) tasks, for one-handed thumb use on touchscreen-based mobile devices without degrading performance and preference. The minimum space between elements is recommended to 2mm ("UX for Mobile: The Rise of Fat-Finger Design").

Regarding the limited motion available to the thumb, Hoober (2017) observed based on collected data that users prefer to view and touch the center of the screen. Figure 9 shows the touch accuracy for the various parts of a mobile phone screen. People prefer reading content at the center of the screen and are better at tapping at the center of the screen. This means touch targets in the center can be smaller - as small as 7mm, while corner target sizes must be about 12mm.

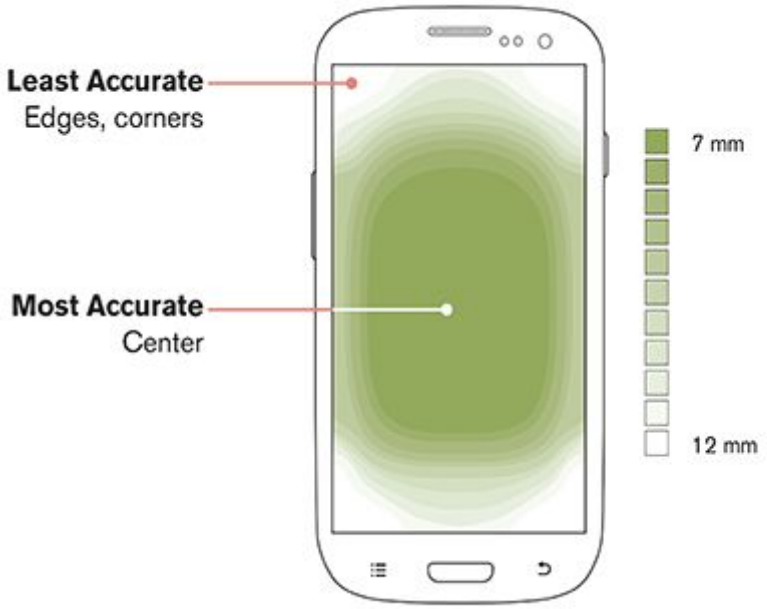


Figure 9. Chart showing touch accuracy for specific parts of the screen

2.4 Use of Color

Effective use of color can enhance communication from screens in human computer interfaces (Marcus, Cowan, & Smith, 1989). Noirhomme-Fraiture and others (2015) recommend to use contrast boundary instead of line boundary, because a contrast boundary is better than a line boundary for making a shape stand out (Easterby, 1970). This signals the importance to avoid outlines but adopt colorful circles with balanced contrast in representing different comments.

Another consideration related to the color usage is the background color. Aleman and other (2018) found that "black text on a white background overstimulates the OFF ganglion cells while white text on black background overstimulates the ON ganglion cells." Thus, pure black and pure white backgrounds tend to cause more eye strain or even vision problems. The situation is even worse for smartphones which cause more serious eye strain and decreased tear production (Smith, 2019). Dark gray is the best option for the majority normal-sighted users to prevent eye strain (UX Movement, 2020).

3. Architecture

The following section describes an overview of the various pages, components and services part of the application. A more detailed walkthrough for each service and component is described in the next section 4. Retrieval, Preprocess and Storage Management.

KANDINSKY Mobile is built using Ionic 4, a complete framework for building hybrid mobile applications. The framework allows development of native-like applications using existing web technologies which also allows for the deployment of a single code base to multiple platforms. The architecture design of KANDINSKY Mobile builds on top of the internal Angular 8 framework to structure the web elements of the application while relying on Ionic Capacitor to access native resources such as native Web Storage.

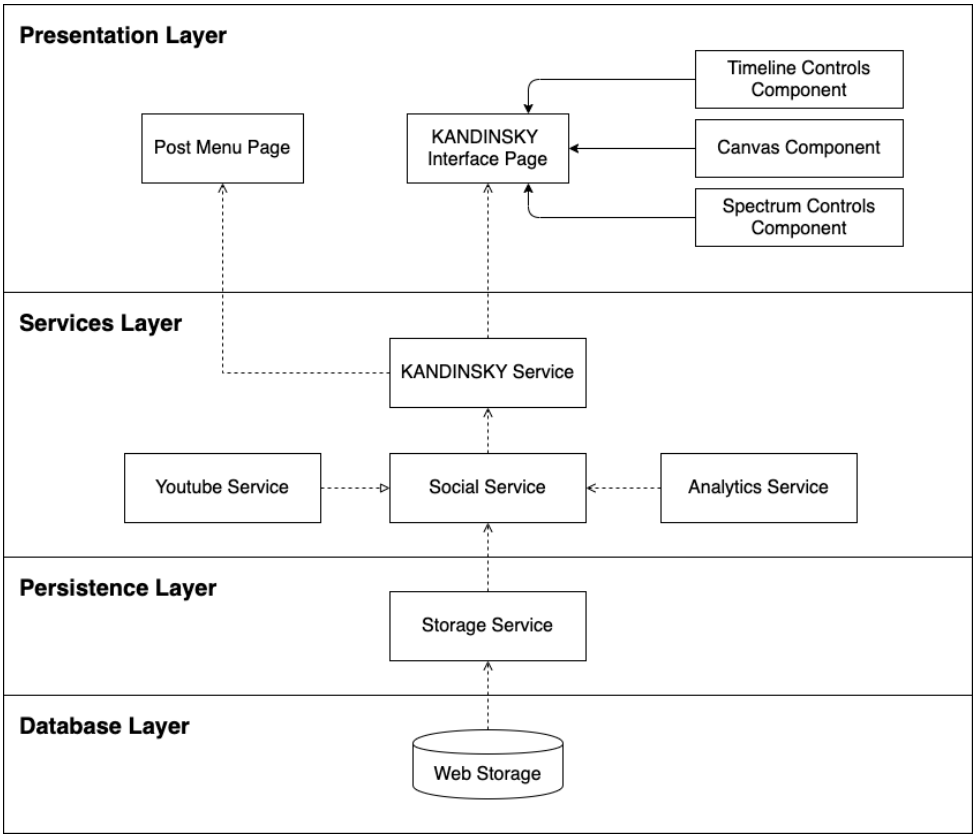


Figure 10: Architecture of Kandinsky Mobile

Figure 10 shows the overall architecture of the application, consisting of services such as the Social Service, the Storage Service, the Analytics Service and the Kandinsky Service. The application also consists of page components such as the Kandinsky Interface Page and the Post Menu Page. These pages internally consist of other components such as the Canvas Component, the Timeline Controls Component and the Spectrum Controls Component.

3.1 Social Service

The Social Service is an abstract service module responsible for retrieving post, comment and comment replies data from social media platforms. The service acts as a provider of helper functions that enable easier integration of other social media platforms. Such helper functions include formalised CRUD interface methods, integration with the built-in Storage Service, and pipelining of post and comment data with their corresponding enhancement methods as provided in the Analytics Service for complete mapping of data into KANDINSKY's SocialPost and SocialComment schema.

Concrete implementations of this module such as Youtube Service implements the actual methods that deal with the exact retrieval of data from the respective platforms, preferably via their officially supported HTTP Application Programming Interface (API). In addition to retrieving data, concrete implementations of Social Service also deal with handling of paginated queries, pre-processing of comment data via recursive hierarchical tree building, and preliminary parsing of post and comment data from their internal schemas into a format that can be utilised by the Social Service when performing data mapping into the standardised schemas.

3.2 Storage Service

The Storage Service is responsible for interfacing other services such as implementations of Social Service with the underlying framework's supported storage APIs. In particular, the service acts as a provider for offline storage using

localForage which wraps web-based storage options like IndexedDB, WebSQL or localStorage into a simplified key-value store API. Concrete implementations of Social Service require acquisition of two separate storage instances to store the post data and comments data separately.

3.3 Analytics Service

The Analytics Service provides utility functions that deal with the enhancement of post and comment data. In particular, the service provides methods to generate topics from a given text and to calculate topic similarity between two modelled topic distributions. When generating topics for textual content, the method utilises a third-party implementation of the Latent Dirichlet Allocation (LDA) (Blei et al., 2003) which is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the textual content are similar. Once the topics are modelled, the same method will also calculate the index intervals of topics generated as used in color highlighting. In addition, the other method which deals with the calculation of topic distribution similarity uses cosine similarity to calculate the distance between two modelled topic distributions.

3.4 Kandinsky Service

The Kandinsky Service is an integral core service which acts as a central component that integrates the UI components with the back-end services. The service injects all implementations of Social Service and wraps them into one unified CRUD API. By wrapping implementations of Social Service into this central service, we are able to minimise the changes needed when supporting new social media platforms in the future. Wrapper functions of this service include the retrieval of all posts from all the Social Service implementations, the routine which handles the retrieval of new social media posts, getting and setting of the current active post and its comments, searching by keyword on the current active post's comments, grouping of comments according to timestamp bins as used in the Spectrum Comment, and other utility methods that are shared across several UI components.

3.5 Kandinsky Interface Page

The Kandinsky Interface Page is the encompassing primary UI component responsible for displaying contents of a stored SocialPost and its list of SocialComment items. The component integrates all internal components together such as the Canvas Component, Timeline Controls Component and the Spectrum Controls Component. In addition, the page also handles the interaction with the Search Bar Component, the cards that displays the list of comments, and other general UI elements such as current timestamp, post title and author display. As the primary UI component, this page integrates the Kandinsky Service to retrieve active post and comments information, and passes them to the respective UI components while managing internal state of all integrations. In addition, the page deals with various event handling operations such as updating of the current timestamp, searching for comments with a given keyword, selection of a concentric circle, spectrum filtering of concentric circles with a given time interval and other general UI-related events.

3.6 Canvas Component

The Canvas Component is the underlying component of Kandinsky Interface Page responsible for the preparation and the actual visualisation of a given set of comment data into concentric circles. The component primarily deals with transformation of comment data into UI-specific data objects, the preparation and updating of the canvas, calculation of concentric circle radii including color assignment and canvas positioning, highlighting of matching concentric circles when searching, handling of zoom controls, managing internal states of each concentric circles and other various internal framework-related tasks. The Canvas Component heavily relies on D3.js, an interactive data-driven JavaScript library for manipulating DOM objects. D3's emphasis on web standards allows deployment of the application to most modern mobile browsers without getting locked into vendor-specific proprietary frameworks.

3.7 Timeline Controls Component

The Timeline Controls Component is the underlying component of Kandinsky Interface Page responsible for the visualisation and updating of the current timestamp which dictates and filters which comments are to be displayed when the application is in Play Mode. The component is a stateful wrapper for Ionic's built-in IonRange Component which is a highly integratable knob slider component. Internally, the component handles control events such as playing and pausing of the player, and the manual setting of timestamp when the knob is dragged by the user. Most of the component settings such as number of circles per step or the play interval in milliseconds are set in the parent Kandinsky Interface Page.

3.8 Spectrum Controls Component

The Spectrum Controls Component is the underlying component of the Kandinsky Interface Page responsible for visualising the distribution of discussion activities over time, and controlling which specific comments to be highlighted given a specific time interval. Internally, the component utilises a hidden IonRange component using its dual-knob feature to allow users to select a time range by dragging the knobs. In visualising the spectrum, this component makes use of D3.js, like the Canvas Component, to visualise the set of bars representing the number of comments published within each bin's time range. To avoid skewed visualisation of the spectrum, a logarithmic scale is used to adjust each bar's vertical height.

3.9 Post Menu Page

The Post Menu Page is the first page the user is directed to when the app is opened. This page provides a list of saved posts so the user can easily load a previously visualised social post, and also an input prompt to allow the user to add a new post by source URL. This component integrates Kandinsky Service to retrieve the list of stored social posts across all supported social platforms and to also set the newly added post as the active post to be displayed in the Kandinsky Interface Page.

4. Retrieval, Preprocess & Storage Management

This section discusses the steps done to facilitate retrieval, preprocessing and storage of data as used in the application. The following examples use Youtube since it's the only supported platform as of the time of this writing.

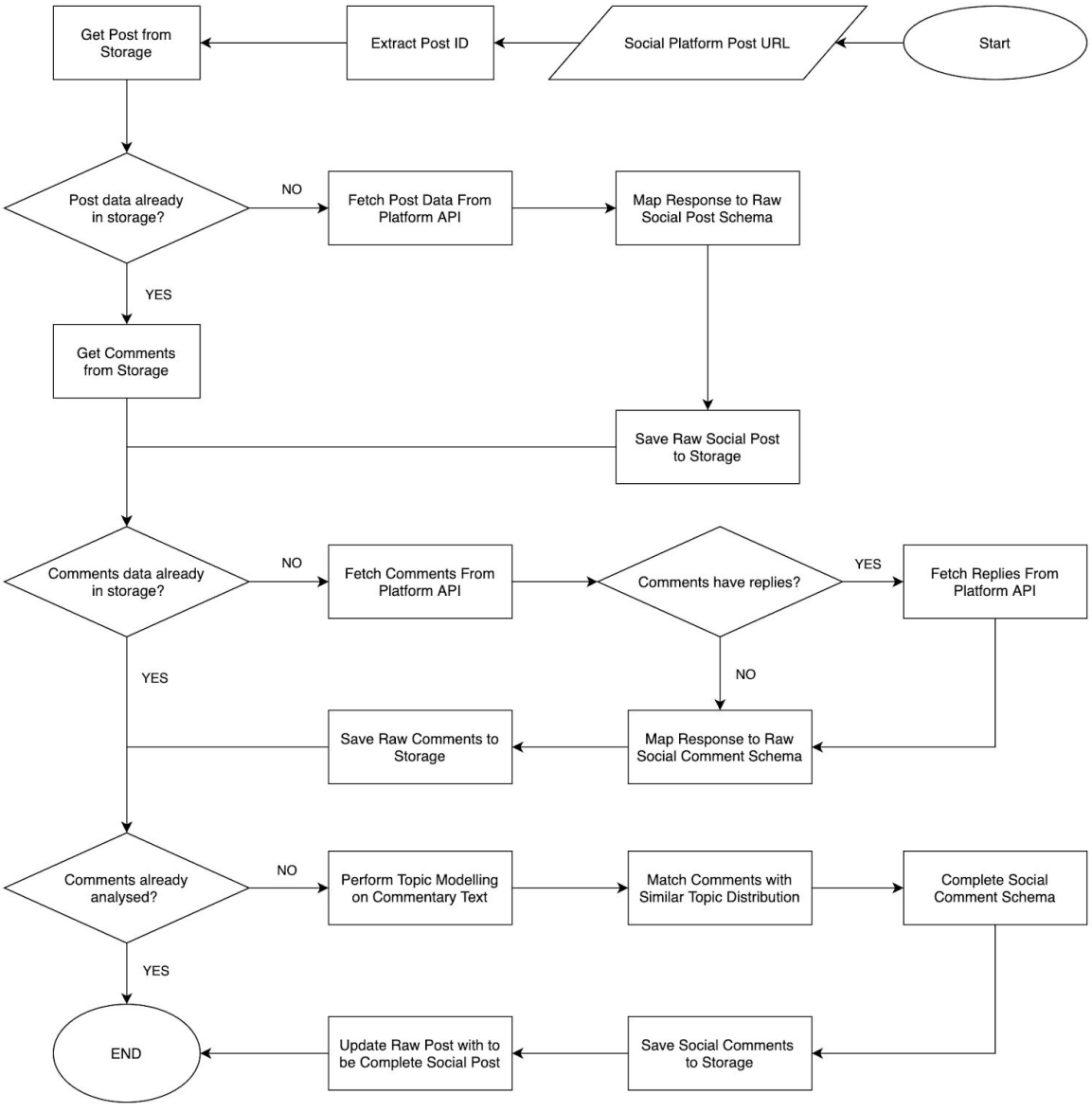


Figure 11: Data Extraction and Representation Overview

The flowchart above describes the general flow of execution from the input of social platform URL all the way to when the data is now set in storage, ready to be visualised by the Canvas Component. Given a URL, the Kandinsky Service through the help of platform-specific Social Services will try to extract the ID of the post the URL points to. When this ID has been identified, the same platform-specific Social Service designated to handle this URL schema will first check if there is an entity with the same ID in its storage cache and if not, will proceed to execute a series of API requests to fetch the general post data, the post's commentary data and the replies to its top level comments if any. For every response received, the same service will map the response into internal schemas designed specifically to work with the other components and services of the application. When this is done, the partially prepared data is saved in storage. Once this partially prepared data is now available, Kandinsky Service will check if the comment data has already been preprocessed and enhanced with analytics metadata. If not, this will call the correct routines to facilitate this enhancement by performing topic modelling and matching the comments with other comments that have similar topic distribution. The enhanced data objects are once again saved to the storage, now with the analytics metadata. At this stage, the data for the post is now ready for visualization.

4.1 Data Retrieval

In the data retrieval process, the implementations of Social Service first check for existing cache instances in local storage, and if not found, proceed to execute a sequence of API requests to fetch the data from their corresponding social media platforms. The sequence of API requests consist of the request to get general post information, paginated requests to get the list of top level comments, and then for each top level comment, the paginated requests to get the replies to it if any.

When the user inputs the post's URL in the Post Menu Page, the call of execution is passed to Kandinsky Service which is responsible for determining the platform of the given URL. Once identified, the ID is extracted and used to retrieve the post's data.

To illustrate this, a video uploaded by NBC New York entitled *President Trump Suggests 'Injecting' Disinfectant as Coronavirus Cure | NBC New York* with the URL "https://www.youtube.com/watch?v=zicGxU5MfwE" is used as the example post. Given this URL, Kandinsky Service will iterate over the implementations of Social Service and check if any of the concrete services is able to parse the URL. Youtube Service's implementation of the *extractId* routine would be called eventually and return the video post's ID ***zicGxU5MfwE*** which is internal to Youtube's systems. This uses Regular Expression patterns to extract the post ID from the URL to make it dynamic and support a wider range of formats that point to the same web resource. Table 2 below exhaustively lists supported Youtube URL schemes with the omission of other URL request parameters.

Video URL	**Post ID**
https://m.youtube.com/watch?v=**zicGxU5MfwE**	zicGxU5MfwE
https://www.youtube.com/watch?v=**zicGxU5MfwE**	zicGxU5MfwE
http://www.youtube.com/watch?v=**zicGxU5MfwE**#t=0m10s	zicGxU5MfwE
http://www.youtube.com/embed/**zicGxU5MfwE**?rel=0	zicGxU5MfwE
http://youtu.be/**zicGxU5MfwE**	zicGxU5MfwE

Table 2: Supported Youtube URL Schemes

The same video ID will be then used as the same internal identification string within the application. Once this post ID is ready, Kandinsky Service's *setActivePost* routine will pass the execution flow to Youtube Service's *getPost* routine which will try to retrieve the post data from the internal storage using the ID and if not found, will proceed to start the web extraction process.

The web extraction process as implemented in Youtube Service's *fetchPost* routine will build the API-compliant request parameters and execute the GET requests to Youtube's servers. As mentioned in the overview, the first request to be made is to first get the video post's general information using the **Videos: list** endpoint. This is used to get the title of the post, a description of the post (if any), the author's name,

and estimates of viewer interaction statistics. The parsing of this response will be discussed in the next section 4.2 Model Mapping. A shortened preview of the request and response is detailed in Table 3 below.

<table>
<tr><td>Method</td><td colspan="2">GET</td></tr>
<tr><td>URL</td><td colspan="2">https://www.googleapis.com/youtube/v3/videos</td></tr>
<tr><td rowspan="3">Parameters</td><td>part</td><td>id,statistics,snippet</td></tr>
<tr><td>key</td><td><YOUTUBE API KEY></td></tr>
<tr><td>id</td><td>zicGxU5MfwE</td></tr>
<tr><td>Response</td><td colspan="2">{
"id": "zicGxU5MfwE",
"snippet": {
"publishedAt": "2020-04-24T00:54:54.000Z",
"title": "President Trump Suggests ‘Injecting’ ...",
"description": "During a press conference ...",
"thumbnails": {
"standard": {
"url": "https://...",
"width": 640,
"height": 480
}
},
"channelTitle": "NBC New York",
},
"statistics": {
"likeCount": "2113",
"commentCount": "4203"
}
}</td></tr>
</table>

Table 3: Videos List Request and Response

Once this response has been mapped to the app’s internal Post model and has been stored in local storage, the same *setActivePost* of Kandinsky Service will proceed to get the comments of the post by passing the flow of execution to Youtube Service’s *getComments* function.

Like the *getPost* routine, the *getComments* function will first check the local storage for any cached comment instances for the post with the given ID. If not found, this will execute Youtube Service's *fetchComments* routine to retrieve the video post's top level comments using Youtube API's **CommentThreads: list** endpoint. This paginated endpoint is used to fetch a list of top level comments of the post including each comment's content, author details, publication timestamp, and some user interaction statistics. Along with this list is the next page's token which is used as an offset to request the next page of comments if any. The parsing of this response will be discussed in the next section 4.2 Model Mapping. A shortened preview of the request and response is detailed in Table 4 below.

Method	GET	
URL	https://www.googleapis.com/youtube/v3/commentThreads	
Parameters	part	snippet
	key	<YOUTUBE API KEY>
	videoId	zicGxU5MfwE
	pageToken	<NEXT PAGE TOKEN>
	maxResults	100
	textFormat	plainText

Response	<pre>{
	"nextPageToken": "QURTSl...",
	"items": [
	{
	“id”: “Ugx2Ck...”,
	“snippet”: {
	“topLevelComment”: {
	“snippet”: {
	“authorDisplayName”: “...”,
	“totalReplyCount”: 4
	},
	“textDisplay”: “This is what people are...",
	“likeCount”: 0,
	"publishedAt": "2020-04-26T14:38:54.000Z"
	}
	}</pre>	

	}, ...] }

Table 4: CommentThreads List Request and Response

In the iteration of top level comment pages, every top level comment is checked if the *snippet.topLevelComment.snippet.totalReplyCount* field is greater than 0. If the value is found to be greater than 0, a child routine of Youtube Service called *fetchReplies* is executed to fetch the paginated replies to the current top level comment. This is done for every top level comment in the current page before proceeding to the next top level comment page.

Unlike the previous API routines, the *fetchReplies* method does not check local storage first for replies to a particular comment since the entire data of a post, including comments and replies to its comments, are all indexed by the post's ID. More will be explained in section 4.4 Data Storage. The fetchReplies method uses Youtube's **Comments: List** endpoint which returns a paginated list of replies to a comment which all resemble a similar schema to a top level comment. Along with this list is the next page's token which is used as an offset to request the next page of replies to the comment if any. The parsing of this response will be done together as a whole with its top level comment. A shortened preview of the request and response is detailed in Table 5 below.

Method	GET	
URL	https://www.googleapis.com/youtube/v3/comments	
Parameters	part	id,snippet
	key	<YOUTUBE API KEY>
	parentId	Ugx2Ck…
	pageToken	<NEXT PAGE TOKEN>
	maxResults	100

	textFormat	plainText
Response	{ "nextPageToken": "QURTSl...", "items": [{ “id”: “Ugx2Ck...”, “snippet”: { “topLevelComment”: { “snippet”: { “authorDisplayName”: “...”, “totalReplyCount”: 0 }, “textDisplay”: “...", “likeCount”: 0, "publishedAt": "2020-04-26T14:38:54.000Z" } } }, ...] }	

Table 5: Comments List Request and Response

The data retrieval process is considered complete once all the required information about a post is extracted: the post’s general information, the post’s top level comments, and the comments which are replies to its top level comments. After this, the model mapping process begins.

4.2 Model Mapping

In the model mapping process, the data retrieved from the social media platform is massaged and reformatted in order to shape it in a schema the other internal components can understand. By mapping the raw data received from the social media platform into a standard schema, we maximize the maintainability of our application since this minimizes the work needed to be done when updating the application to support a different social media platform. By adding this model mapping layer to the

pipeline, we enforce the independence of the other services from any concrete Social Service.

By contract, the implementations of Social Service would have to map the data they get from the API into preliminary models referred to as *RawSocialPost* and *RawSocialComment*. The *Raw* prefix indicates that this is not the final product of the *getPost* and *getComments* routines of SocialService as they still need to be processed by the Analytics Service. With reference to Youtube Service, Table 6 below details the internal schemas used to represent the preliminary *RawSocialPost* model.

Field	**Type**	**Description**
id	string	The video ID string from Youtube.
content	string	The description of the uploaded video.
imageUrl	string	The thumbnail of the video.
authorName	string	The name of the channel author.
publishTimestamp	number	The UNIX timestamp of the date of upload.
likeCount	number	The number of likes.
commentCount	number	The estimated number of comments.
raw	object	The raw data retrieved from Youtube.
platform	string	A string to denote the platform “Youtube”.
sourceUrl	string	The platform URL of the video.

Table 6: RawSocialPost Schema

In mapping the post’s comments into a *RawSocialComment*, there is a need to transform the comments retrieved from Youtube into a tree-like data structure in order to represent the hierarchical nesting of social media discussions. That is, to represent the fact that a reply to a top level comment may not necessarily be in response to the top level comment itself, but in response to a reply to it. This step is necessary because Youtube’s schema only presents the data in a 2-dimension data

structure i.e. list of top level comments and a list of replies to each top level comment.

Since Youtube's default Comment schema does not provide a field to represent this nested hierarchy, the comment mapping routine of Youtube Service uses an iterative subroutine called *buildCommentTree* which reduces the flat list of replies to a comment into a nested object data structure where the head of the tree is the top level comment and the nodes are the replies to the comment node above it. Internal to this routine is an algorithm to detect whether a comment is a reply to previously seen unreplied comment. This uses Regular Expression patterns to match whether the comment's content starts with a tag that references a previously seen comment's author name.

The process of building the hierarchy of comments is done together with the mapping of comment data from Youtube into the internal *RawSocialComment* schema. With reference to Youtube Service, Table 7 below details the internal schemas used to represent the tree-enabled preliminary *RawSocialComment* model.

Field	**Type**	**Description**
id	string	The comment ID string from Youtube.
content	string	The comment content as written by the author.
imageUrl	string	The profile picture URL of the author.
authorName	string	The name of the comment author.
publishTimestamp	number	The UNIX timestamp of the date of comment publication.
likeCount	number	The number of likes.
commentCount	number	The number of replies.
raw	object	The raw data retrieved from Youtube.

postId	string	The original video post's ID.
parentId	string	The ID of the parent comment it replies to. NULL if it is a top level comment.
parentAuthorName	string	The parent comment's author name.
comments	*RawSocialComment[]*	A list of RawSocialComment to represent the list of replies as built by *buildCommentTree*.

Table 7: RawSocialComment Schema

These transformed data, although preliminary in nature, are transiently stored in the data storage so as to commit the expensive processing efforts done so far. By having this checkpoint, we cache the data prepared and protect the pre-processed data from any exceptions that happen beyond this point. That is to say, even if there was an error in the next step of data processing, when the user loads the same video again, there would be no need to perform the entire data retrieval process again since it would fetch this from the local storage instead. A detailed explanation of the storage mechanism will be described in section 4.4 Data Storage.

4.3 Data Preprocessing

Once the post data has been transformed and saved into storage as *RawSocialPost* and the corresponding comments as a tree of *RawSocialComment* list, the data is passed to Social Service's *analyzeComments routine* which facilitates the enhancements needed to finally transform the data into the complete *SocialPost* and *SocialComment* schemas. The execution of this step is done with the help of an injected Analytics Service. The Analytics Service only works with raw strings and simpler data structures disjoint from the model schemas described in the previous section 4.2 Model Mapping. This design is intentional to keep the Analytics Service loosely coupled from the other services.

In the *analyzeComments* function, the data is first retrieved from the storage as prepared in the Model Mapping phase. For each comment, topics are generated using Analytics Service's *generateTopics* routine which uses Latent Dirichlet Allocation (LDA) to statistically identify the prominent topics. Each topical string identified is returned together with a metadata object containing the list of index pairs used to identify where in the original comment content the string is located, and a probabilistic value to denote the likeliness of this topic for this comment. For example, given the string "Cats are small. Dogs are big. Cats like to chase mice. Dogs like to eat bones.", the *generateTopics* routine will return an object containing the topics with their metadata as detailed in Table 8 below.

Topic	**Metadata**
cats	{ "indices": [[0,4], [30,34]], "probability": 0.0021 }
dogs	{ "indices": [[16,20], [55,59]], "probability": 0.0019 }
small	{ "indices": [[10,15]], "probability": 0.0010 }
mice	{ "indices": [[49,53]], "probability": 0.0010 }
chase	{ "indices": [[44,48]], "probability": 0.0010 }

Table 8: Topic Metadata Example

After the topics for each comment has been generated, another iteration over all the comments will be executed to then find other comments that contain a similar tropic distribution with the current comment in the iteration using cosine similarity. In the default setting, only comments that yield a similarity score greater than 0.5 are considered *similar* to a comment. Any score less than that will not be considered similar enough and will be ignored.

For each comment, the list of topics and a list of similar comments references are embedded within the comment itself in a new field called *analytics.similarity*. By embedding this analysis result into the comment itself, we avoid redundancy in our data storage. After the comments have been enhanced, the parent post itself will be retrieved from storage and updated its new *metadata* field with new information such

as total actual count of comments, the timestamp of the first and last comments, and other internally used flags. Collectively this process will perform the final mapping of data from *RawSocialPost* and *RawSocialComment* to the complete *SocialPost* and *SocialComment* respectively. The following tables describe the new fields added to the models after the data preprocessing phase.

Field	**Type**	**Description**
metadata.archived	boolean	Flag whether to delete the post or not. Currently not in use.
metadata.archiveTimestamp	number	UNIX timestamp to record when the post was archived. Currently not in use.
metadata.createTimestamp	number	UNIX timestamp to record when the post was fetched. Set when first fetched.
metadata.lastUpdateTimestamp	number	UNIX timestamp to record when the post was updated. Set when first fetched and later after pre-processing.
metadata.comments.actualCount	number	Counts the actual number of comments retrieved. May differ from Youtube's count due to various reasons.
metadata.comments.lastUpdateTimestamp	number	UNIX timestamp to record when the comments for this post was fetched and saved. Currently not in use.
metadata.comments.lastAnalysisTimestamp	number	UNIX timestamp to record when the analysis process for comments was executed. This is also used as a flag to determine whether the comments have already been analysed or not.
metadata.comments.firstTimestamp	number	UNIX timestamp to hold the first comment's publish timestamp.
metadata.comments.lastTimestamp	number	UNIX timestamp to hold the last comment's publish timestamp.

Table 9: Field Updates to RawSocialPost to yield SocialPost Schema

Field	Type	Description
comments	*SocialComment[]*	Updated list of SocialComment.
analytics.similarity .comments	*SimilarComment[]*	List of SimilarComment objects containing the score and the comment ID.
analytics.similarity .topics	{string: TopicMetadata}	Contains the topic distribution generated for the comment content.

Table 10: Field Updates to RawSocialComment to yield SocialComment Schema

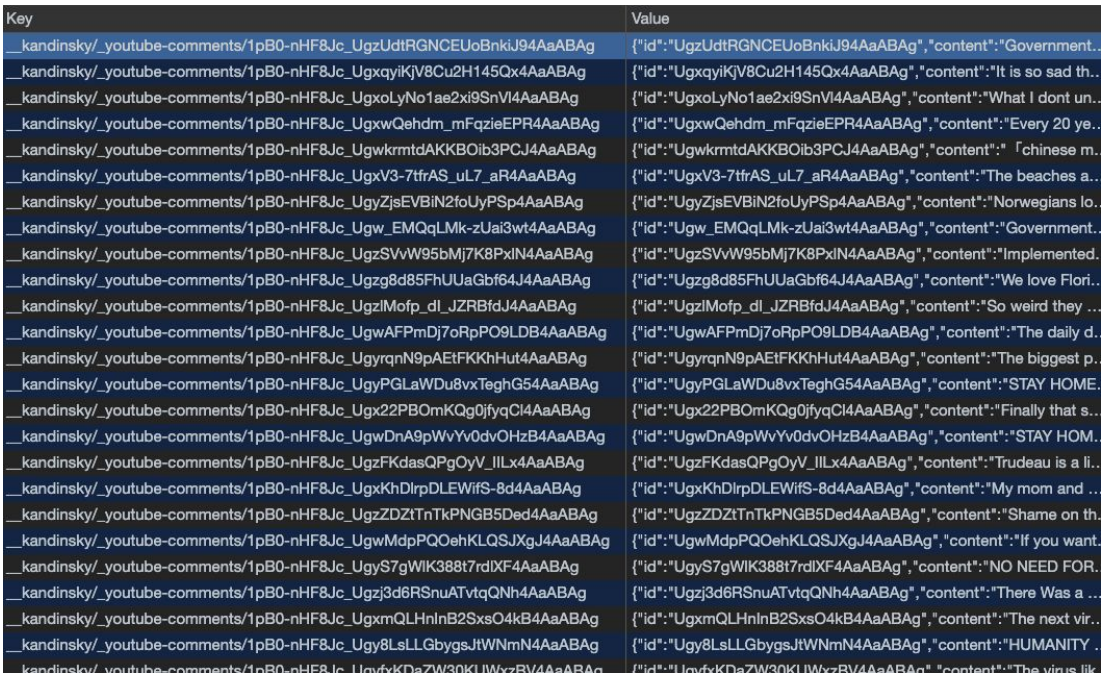
4.4 Data Storage

The internal storage mechanism of the application uses a library called localForage which wraps offline web-based storage options into a simplified key-value store API. Given the need to support multiple kinds of mobile devices with limited storage options available, it is considered critical to use such cross-platform support which would automatically fallback the web storage driver depending on which drivers are supported by the device. By default, localForage will attempt to use a database driver in the sequence: IndexedDB, WebSQL and localStorage. All three database drivers support the same key-value document storage pattern, which makes it easier to support devices with limited options.

Storing SocialPost and SocialComment entities are done iteratively throughout the extraction and analysis processes. As discussed in the previous sections, data is stored transiently and is incrementally updated as the data preparation is being done. For SocialPost objects, it is stored and indexed by the post's ID. That means the same post can also be retrieved by simply querying the Storage Service's *getItem* method by passing the post's ID to retrieve the SocialPost object. Similarly, for SocialComment objects, they are stored and indexed by a key derived as an aggregation of the post's ID and the comment's ID.

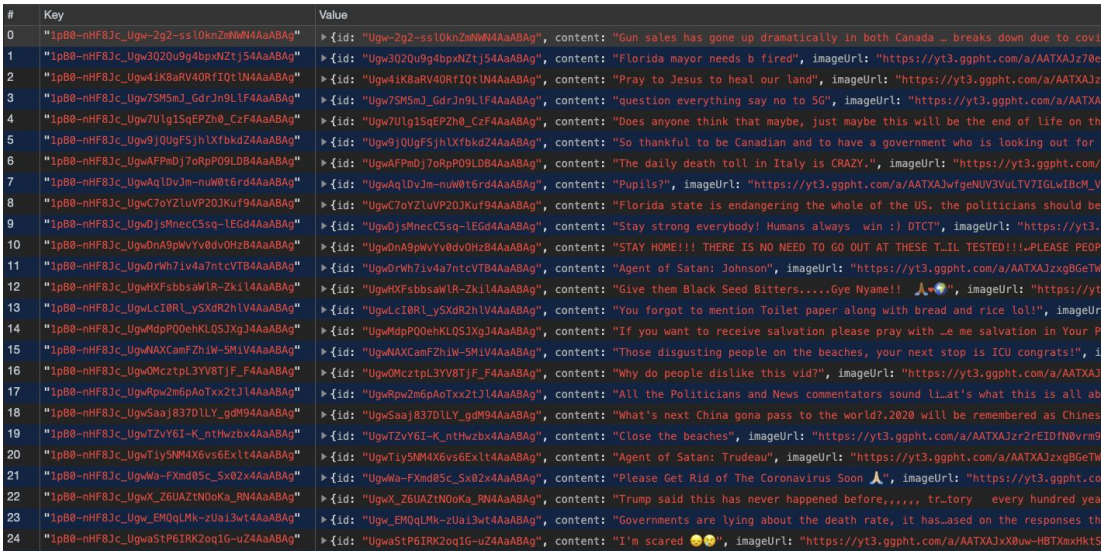
For example, for a comment with ID *AAAAAA* of a social post with ID *BBBBBBB*,

the key to be used for this storage entity would be *BBBBBBB_AAAAAA*. By prepending the comment item's storage key with the post's ID, we are able to retrieve all comments that belong to a particular post. Note that only top level comments are indexed this way and replies to those top-level comments are embedded within their own top-level comment object.



Key	Value
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgzUdtRGNCEUoBnkiJ94AaABAg	{"id":"UgzUdtRGNCEUoBnkiJ94AaABAg","content":"Government..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgxqyiKjV8Cu2H145Qx4AaABAg	{"id":"UgxqyiKjV8Cu2H145Qx4AaABAg","content":"It is so sad th..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgxoLyNo1ae2xi9SnVl4AaABAg	{"id":"UgxoLyNo1ae2xi9SnVl4AaABAg","content":"What I dont un..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgxwQehdm_mFqzieEPR4AaABAg	{"id":"UgxwQehdm_mFqzieEPR4AaABAg","content":"Every 20 ye..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgwkrmtdAKKBOib3PCJ4AaABAg	{"id":"UgwkrmtdAKKBOib3PCJ4AaABAg","content":"「chinese m..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgxV3-7tfrAS_uL7_aR4AaABAg	{"id":"UgxV3-7tfrAS_uL7_aR4AaABAg","content":"The beaches a..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgyZjsEVBiN2foUyPSp4AaABAg	{"id":"UgyZjsEVBiN2foUyPSp4AaABAg","content":"Norwegians lo..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_Ugw_EMQqLMk-zUai3wt4AaABAg	{"id":"Ugw_EMQqLMk-zUai3wt4AaABAg","content":"Government..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgzSVvW95bMj7K8PxlN4AaABAg	{"id":"UgzSVvW95bMj7K8PxlN4AaABAg","content":"Implemented.
_kandinsky/_youtube-comments/1pB0-nHF8Jc_Ugzg8d85FhUUaGbf64J4AaABAg	{"id":"Ugzg8d85FhUUaGbf64J4AaABAg","content":"We love Flori..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgzlMofp_dI_JZRBfdJ4AaABAg	{"id":"UgzlMofp_dI_JZRBfdJ4AaABAg","content":"So weird they ..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgwAFPmDj7oRpPO9LDB4AaABAg	{"id":"UgwAFPmDj7oRpPO9LDB4AaABAg","content":"The daily d..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgyrqnN9pAEtFKKhHut4AaABAg	{"id":"UgyrqnN9pAEtFKKhHut4AaABAg","content":"The biggest p..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgyPGLaWDu8vxTeghG54AaABAg	{"id":"UgyPGLaWDu8vxTeghG54AaABAg","content":"STAY HOME..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_Ugx22PBOmKQg0jfyqCl4AaABAg	{"id":"Ugx22PBOmKQg0jfyqCl4AaABAg","content":"Finally that s..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgwDnA9pWvYv0dvOHzB4AaABAg	{"id":"UgwDnA9pWvYv0dvOHzB4AaABAg","content":"STAY HOM..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgzFKdasQPgOyV_IILx4AaABAg	{"id":"UgzFKdasQPgOyV_IILx4AaABAg","content":"Trudeau is a li..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgxKhDlrpDLEWifS-8d4AaABAg	{"id":"UgxKhDlrpDLEWifS-8d4AaABAg","content":"My mom and ..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgzZDZtTnTkPNGB5Ded4AaABAg	{"id":"UgzZDZtTnTkPNGB5Ded4AaABAg","content":"Shame on th..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgwMdpPQOehKLQSJXgJ4AaABAg	{"id":"UgwMdpPQOehKLQSJXgJ4AaABAg","content":"If you want..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgyS7gWlK388t7rdlXF4AaABAg	{"id":"UgyS7gWlK388t7rdlXF4AaABAg","content":"NO NEED FOR..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_Ugzj3d6RSnuATvtqQNh4AaABAg	{"id":"Ugzj3d6RSnuATvtqQNh4AaABAg","content":"There Was a ..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_UgxmQLHnlnB2SxsO4kB4AaABAg	{"id":"UgxmQLHnlnB2SxsO4kB4AaABAg","content":"The next vir..
_kandinsky/_youtube-comments/1pB0-nHF8Jc_Ugy8LsLLGbygsJtWNmN4AaABAg	{"id":"Ugy8LsLLGbygsJtWNmN4AaABAg","content":"HUMANITY ..

Figure 12: Comments as Stored in an Emulated Browser's Local Storage



#	Key	Value
0	"1pB0-nHF8Jc_Ugw-2g2-sslOknZmNW4AaABAg"	▸{id: "Ugw-2g2-sslOknZmNW4AaABAg", content: "Gun sales has gone up dramatically in both Canada … breaks down due to covi
1	"1pB0-nHF8Jc_Ugw3Q2Qu9g4bpxNZtj54AaABAg"	▸{id: "Ugw3Q2Qu9g4bpxNZtj54AaABAg", content: "Florida mayor needs b fired", imageUrl: "https://yt3.ggpht.com/a/AATXAJz70e
2	"1pB0-nHF8Jc_Ugw4iK8aRV4ORfIQtlN4AaABAg"	▸{id: "Ugw4iK8aRV4ORfIQtlN4AaABAg", content: "Pray to Jesus to heal our land", imageUrl: "https://yt3.ggpht.com/a/AATXAJz
3	"1pB0-nHF8Jc_Ugw7SM5mJ_GdrJn9LlF4AaABAg"	▸{id: "Ugw7SM5mJ_GdrJn9LlF4AaABAg", content: "question everything say no to 5G", imageUrl: "https://yt3.ggpht.com/a/AATXA
4	"1pB0-nHF8Jc_Ugw7Ulg1SqEPZh0_CzF4AaABAg"	▸{id: "Ugw7Ulg1SqEPZh0_CzF4AaABAg", content: "Does anyone think that maybe, just maybe this will be the end of life on th
5	"1pB0-nHF8Jc_Ugw9jQUgFSjhlXfbkdZ4AaABAg"	▸{id: "Ugw9jQUgFSjhlXfbkdZ4AaABAg", content: "So thankful to be Canadian and to have a government who is looking out for
6	"1pB0-nHF8Jc_UgwAFPmDj7oRpPO9LDB4AaABAg"	▸{id: "UgwAFPmDj7oRpPO9LDB4AaABAg", content: "The daily death toll in Italy is CRAZY.", imageUrl: "https://yt3.ggpht.com/
7	"1pB0-nHF8Jc_UgwAqlDvJm-nuW0t6rd4AaABAg"	▸{id: "UgwAqlDvJm-nuW0t6rd4AaABAg", content: "Pupils?", imageUrl: "https://yt3.ggpht.com/a/AATXAJwfgeNUV3VuLTV7IGLwIBcM_V
8	"1pB0-nHF8Jc_UgwC7oYZluVP2OJKuf94AaABAg"	▸{id: "UgwC7oYZluVP2OJKuf94AaABAg", content: "Florida state is endangering the whole of the US. the politicians should be
9	"1pB0-nHF8Jc_UgwDjsMnecC5sq-lEGd4AaABAg"	▸{id: "UgwDjsMnecC5sq-lEGd4AaABAg", content: "Stay strong everybody! Humans always win :) DTCT", imageUrl: "https://yt3.
10	"1pB0-nHF8Jc_UgwDnA9pWvYv0dvOHzB4AaABAg"	▸{id: "UgwDnA9pWvYv0dvOHzB4AaABAg", content: "STAY HOME!!! THERE IS NO NEED TO GO OUT AT THESE T…IL TESTED!!!↵PLEASE PEOP
11	"1pB0-nHF8Jc_UgwDrWh7iv4a7ntcVTB4AaABAg"	▸{id: "UgwDrWh7iv4a7ntcVTB4AaABAg", content: "Agent of Satan: Johnson", imageUrl: "https://yt3.ggpht.com/a/AATXAJzxgBGeTW
12	"1pB0-nHF8Jc_UgwHXFsbbsaWlR-Zkil4AaABAg"	▸{id: "UgwHXFsbbsaWlR-Zkil4AaABAg", content: "Give them Black Seed Bitters.....Gye Nyame!! 🙏🌍", imageUrl: "https://yt
13	"1pB0-nHF8Jc_UgwLcI0Rl_ySXdR2hlV4AaABAg"	▸{id: "UgwLcI0Rl_ySXdR2hlV4AaABAg", content: "You forgot to mention Toilet paper along with bread and rice lol!", imageUr
14	"1pB0-nHF8Jc_UgwMdpPQOehKLQSJXgJ4AaABAg"	▸{id: "UgwMdpPQOehKLQSJXgJ4AaABAg", content: "If you want to receive salvation please pray with …e me salvation in Your P
15	"1pB0-nHF8Jc_UgwNAXCamFZhiW-5MiV4AaABAg"	▸{id: "UgwNAXCamFZhiW-5MiV4AaABAg", content: "Those disgusting people on the beaches, your next stop is ICU congrats!", i
16	"1pB0-nHF8Jc_UgwOMcztpL3YV8TjF_F4AaABAg"	▸{id: "UgwOMcztpL3YV8TjF_F4AaABAg", content: "Why do people dislike this vid?", imageUrl: "https://yt3.ggpht.com/a/AATXAJ
17	"1pB0-nHF8Jc_UgwRpw2m6pAoTxx2tJl4AaABAg"	▸{id: "UgwRpw2m6pAoTxx2tJl4AaABAg", content: "All the Politicians and News commentators sound li…at's what this is all ab
18	"1pB0-nHF8Jc_UgwSaaj837DlLY_gdM94AaABAg"	▸{id: "UgwSaaj837DlLY_gdM94AaABAg", content: "What's next China gona pass to the world?.2020 will be remembered as Chines
19	"1pB0-nHF8Jc_UgwTZvY6I-K_ntHwzbx4AaABAg"	▸{id: "UgwTZvY6I-K_ntHwzbx4AaABAg", content: "Close the beaches", imageUrl: "https://yt3.ggpht.com/a/AATXAJzr2rEIDfN0vrm9
20	"1pB0-nHF8Jc_UgwTiy5NM4X6vs6Exlt4AaABAg"	▸{id: "UgwTiy5NM4X6vs6Exlt4AaABAg", content: "Agent of Satan: Trudeau", imageUrl: "https://yt3.ggpht.com/a/AATXAJzxgBGeTW
21	"1pB0-nHF8Jc_UgwWa-FXmd05c_Sx02x4AaABAg"	▸{id: "UgwWa-FXmd05c_Sx02x4AaABAg", content: "Please Get Rid of The Coronavirus Soon 🙏", imageUrl: "https://yt3.ggpht.co
22	"1pB0-nHF8Jc_UgwX_Z6UAZtNOoKa_RN4AaABAg"	▸{id: "UgwX_Z6UAZtNOoKa_RN4AaABAg", content: "Trump said this has never happened before,,,,,, tr…tory every hundred yea
23	"1pB0-nHF8Jc_Ugw_EMQqLMk-zUai3wt4AaABAg"	▸{id: "Ugw_EMQqLMk-zUai3wt4AaABAg", content: "Governments are lying about the death rate, it has…ased on the responses th
24	"1pB0-nHF8Jc_UgwaStP6IRK2oq1G-uZ4AaABAg"	▸{id: "UgwaStP6IRK2oq1G-uZ4AaABAg", content: "I'm scared 😨😲", imageUrl: "https://yt3.ggpht.com/a/AATXAJxX0uw-HBTXmxHktS

Figure 13: Comments as Stored in an Emulated Browser's IndexedDB

5. Circles Layout

This section discusses the design principles and algorithm for the circles layout, which is also evaluated and compared with several baselines. Essentially, KANDINSKY's circles layout arrangement is a custom force-directed simulation arrangement typically used in network layouts but with the exception of intentionally not rendering the edges between concentric circles and also with support for dynamically sized concentric circles using additional forces. A detailed walkthrough of the algorithm to build the layout is described in section 5.2 Algorithm.

KANDINSKY Mobile departs from the traditional hierarchical view by visualizing social discussions into a set of concentric circles. The application of visualizing discussions into concentric circles (Figure 14) takes inspiration from several paintings of renowned Russian painter and art theorist Wassily Kandinsky.

In particular, one of his paintings published in 1926 entitled "Several Circles" illustrates how simple geometric shapes like circles with varying radii can be arranged in an abstract manner against a dark canvas. KANDINSKY Mobile employs a similar approach to visualize the hierarchical relationship that exists within social discussion threads. Each top level comment along with replies to it is represented as a colorful concentric circle called a Nucleus. Social discussions that stem from the nucleus are separately illustrated into surrounding concentric circles termed as Peripherals. Peripherals are often created when social discussion comment threads deviate away from the topics of their respective parental comments.

In addition, his other painting published in 1913 entitled "Color Study: Squares with Concentric Circles" explores how elementary geometrical shapes painted in different color combinations can be used to express his creativity. KANDINSKY Mobile uses the design of a concentric circle to represent the hierarchical structure within a social discussion comment thread. Each comment is represented as a circle wherein its radius represents the number of likes and its position in the sequence of concentricity represents its chronological order within the discussion thread. The colors are used to

distinguish the author of each comment, hence threads with more commentators involved are often drawn in larger radii and more colorful circles.

Figure 14: Visualizing Discussions into Concentric Circles

5.1 Design Principles

5.1.1 Fat-Thumb Friendliness

As mentioned in the literature review section, to ensure fat-thumb friendliness in the application, the target sizes should be at least 9.2 mm in width, and at least 2mm apart from other targets. This is easy to achieve for icons such as the search button. However, the most important clickable elements - concentric circles representing comments - are too many to be fit into the recommended size. The solution is to support the zoom function such that the user can zoom the canvas into a specific

area, and click on the concentric circle the user is interested in. That said, to introduce a similar concept of separation between concentric circle targets, a padded collision force of 30px (7.94mm) is added to the force-directed simulation layout to make the concentric circles push each other, avoiding overlaps and providing adequate space for fat-thumb support.

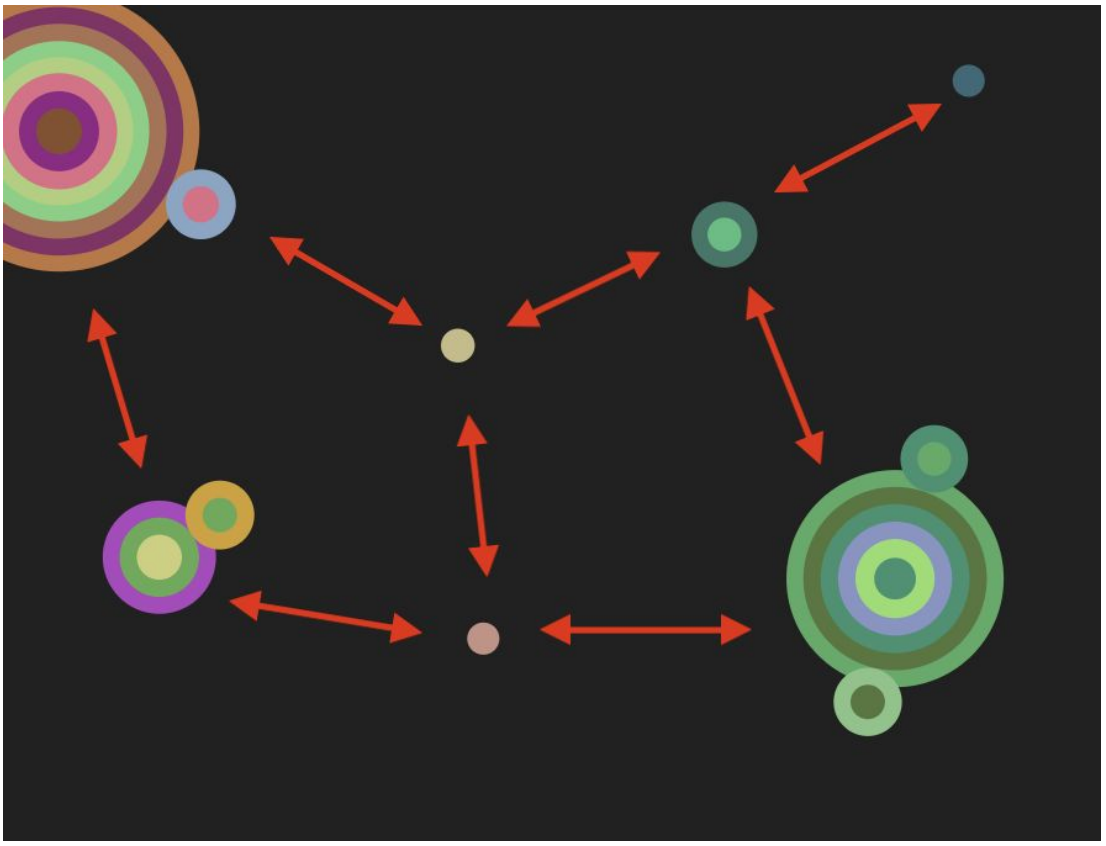


Figure 15: Padded Collision Force between Concentric Circles

5.1.2 Gestalt Principles

Johnson (2014) states that people's vision is designed to see structure, thus making structural forms important for UI design. He also mentions that Gestalt principles (Wertheimer, 1938) are significant for describing human visual perception, thus being able to greatly improve the aesthetics, functionality and user-friendliness of UI design. There are six Gestalt principles, which can be applied in the design of the application - proximity, similarity, continuity, closure, symmetry and common fate,

(Chapman, 2018). Appropriate applications of some of these principles are described below.

The proximity principle states that “elements close together are perceived as a single group or chunk, and are interpreted as being more related than elements that are farther apart” (Lidwell et al., 2010, p.196). Applying this to UX design, related things can be put closer together, with space in between each group, such that the viewer will immediately pick up on the organization and structure intended. Thus, the circles layout in KANDINSKY is designed in a way that concentric circles representing related comments are placed close to each other (i.e. concentric circle representing top level comment and its surrounding peripheral concentric circles representing social discussions stemming from that specific parental comment), while circles representing comments irrelevant to each other are placed apart with gaps in between.

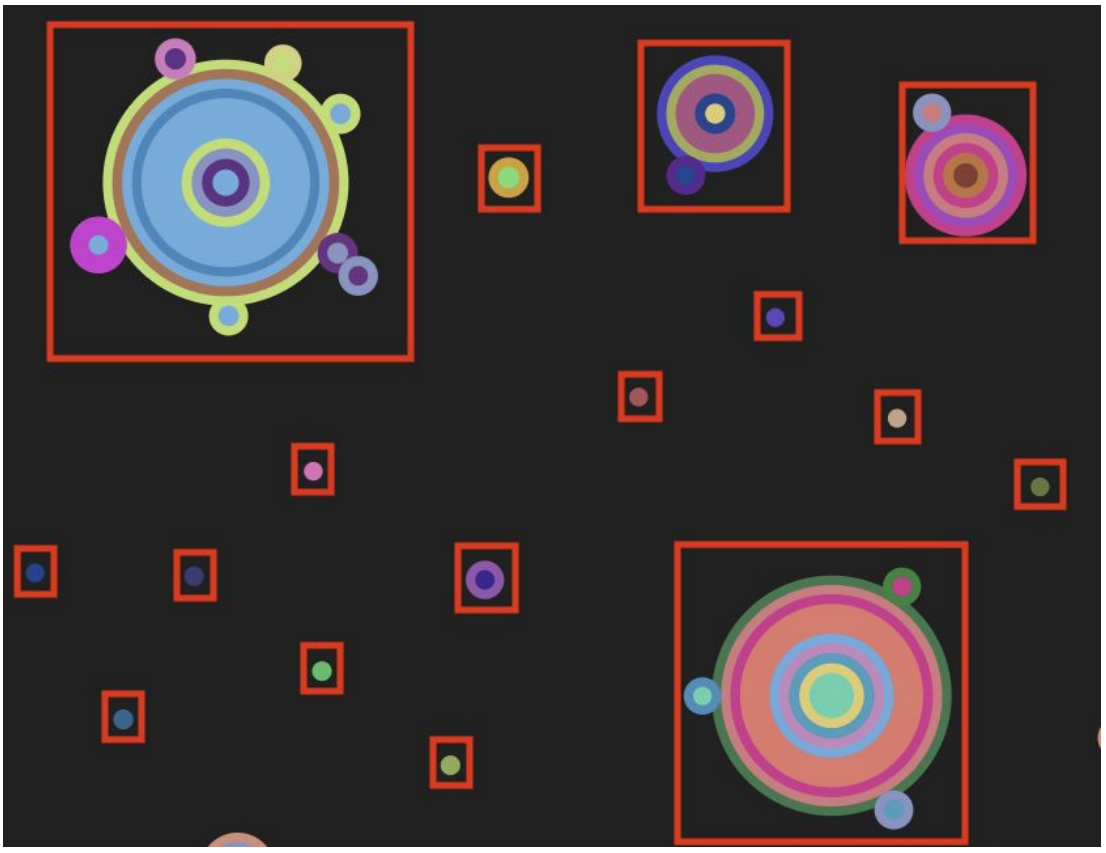


Figure 16: Clustering of Comments into Concentric Circles with Peripherals

According to the similarity principle, “similar elements are perceived as a single group or chunk, and are interpreted as being more related than dissimilar elements”. Similarity can be based on different features – color, shape or size (Lidwell et al., 2010, p. 226). In UX design, similarity is frequently used with other principles such as Proximity to show grouping (Johnson, 2014). In contrast, making things dissimilar allows them to be highlighted and perceived to be more important. Applying this principle, the sizes of the circles appearing on the canvas denotes the number of likes collected by the comment. Furthermore, top-level comments with a lot of replies to it will make the entire concentric circle appear larger. Using this size dimension, the user is quickly able to see which comment threads have similar activity and which comment threads are more or less active than the other.

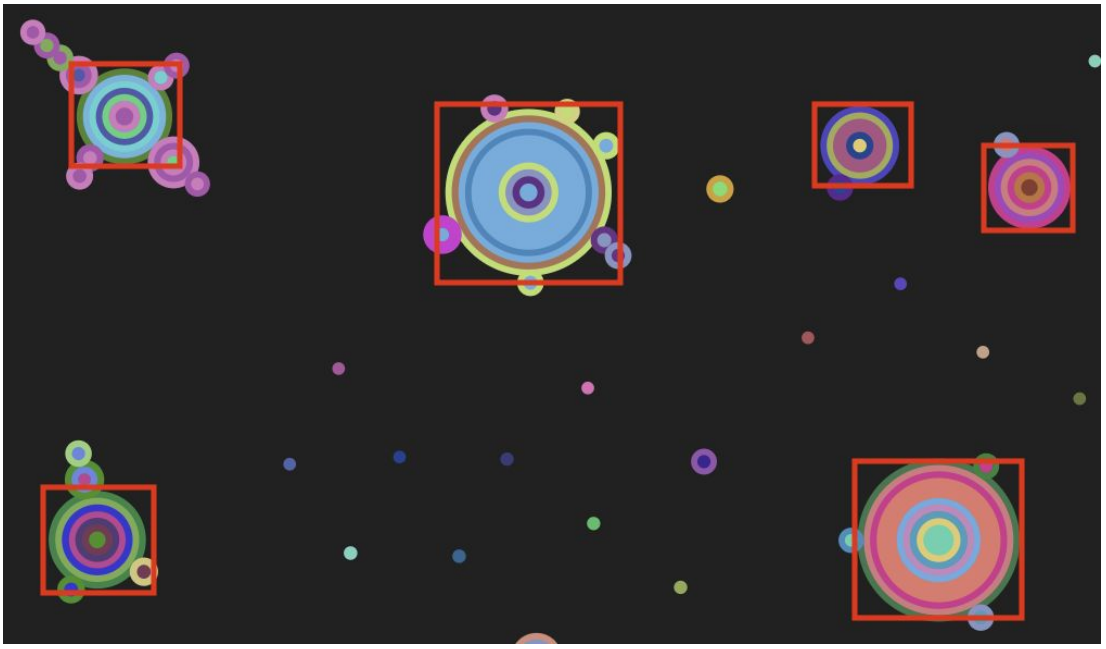


Figure 17: Larger Concentric Circles Denoting Discussions with High User Activity

The principle of continuity suggests that “our visual perception is biased to perceive continuous forms rather than disconnected segments” (Johnson, 2014, p. 18). This continuation can be a valuable tool in UX design when the goal is to establish a linked relationship between visual entities. In the context of the application, this principle is applied by placing peripheral concentric circles within close proximity to its parent concentric circle. This positioning allows the user to perceive the continued conversation within a social comment discussion thread. This phenomenon of snake-like trail of concentric circles is particularly seen on social comment discussion threads between the same 2 users since each reply to a comment would spawn a child peripheral concentric circle.

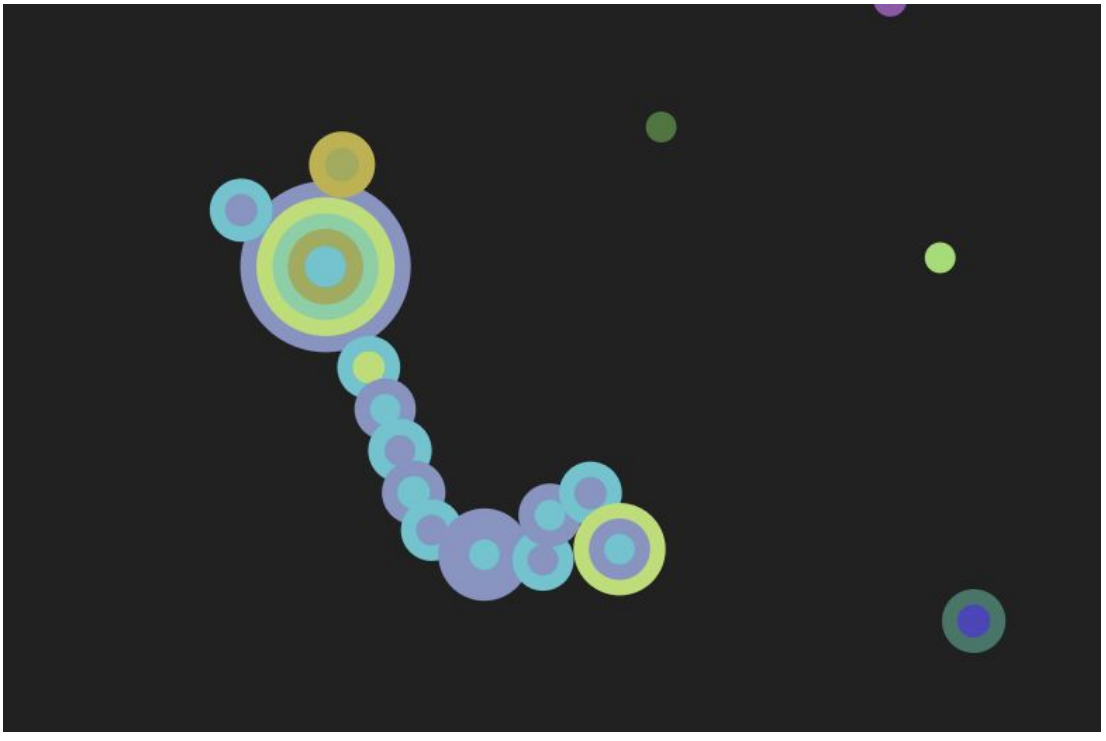


Figure 18: Snake-like Trail of Concentric Circles

The principle of closure means that human brains tend to fill in the missing parts of a design or image to create a whole. This principle states that when users are given a complex arrangement of visual elements, they tend to look for a single, recognizable pattern. This principle is applied to the application by arranging the circles using force simulation to emulate a centered-gravity effect, forcing the circles to collectively form a recognizable shape as a whole. By positioning them in a closed-state, the user is able to navigate quicker since the user is now aware of the boundaries of where the concentric circles are positioned. For example, when the user recognizes the edge of the collective shape, the user will intuitively recognize that this is the border of the canvas, discouraging him/her from panning the canvas further beyond the boundaries.

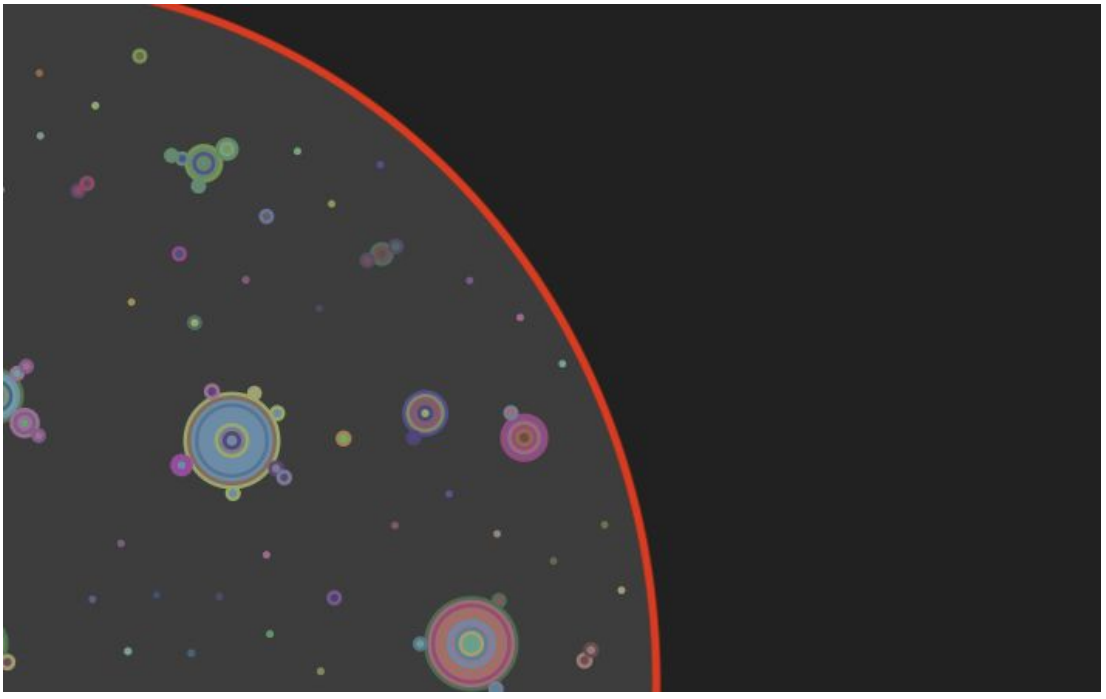


Figure 19: Overlay of the Landscape Boundary

The principle of symmetry suggests that the human vision “automatically organizes and interprets the data so as to simplify it and give it symmetry” (Johnson, 2014, p. 20). Stone (2005) proposes that people “tend to perceive regions bounded by symmetrical borders as coherent figures”. In the application, by choosing a symmetrically circular shape to embody the entire conversation landscape, instead of choosing an arbitrarily shaped structure as it would have been in a network view, it allows the user to automatically perceive that the individual concentric circles are part of the same conversation landscape.

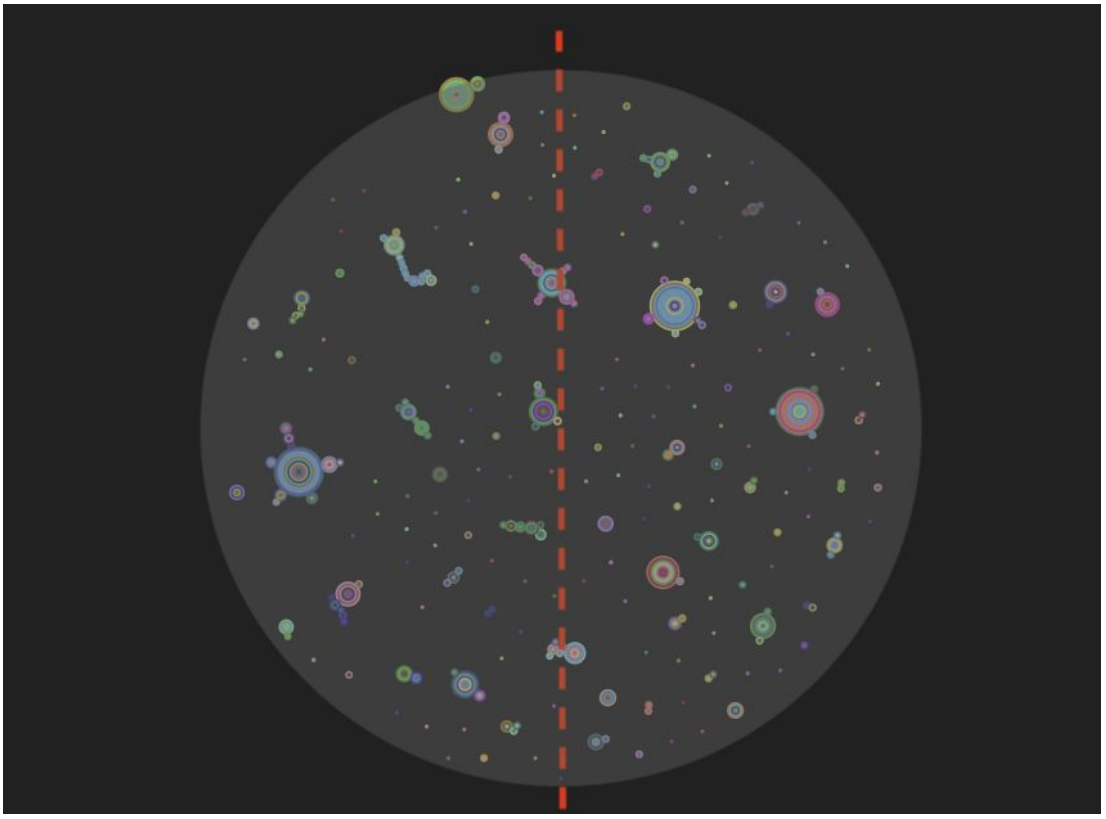


Figure 20: Conversation Landscape Arranged as a Circle

5.2 Algorithm

This subsection describes in detail the procedures done to get the canvas ready to draw the comments as Circles and Concentric Circles. The exact procedure which handles the calculation of each circle's coordinates on the canvas as a cartesian plane is omitted from this report since it is performed internally within D3's library itself. The simulation forces that influence the value of these coordinates are lightly described in subsection 5.2.2. Force-Directed Simulation Layout.

5.2.1 Canvas Preparation

Before the post comments extracted in section 4. Retrieval, Preprocess & Storage Management can be visualised, there is a need to map the SocialComment objects into Circle Datum and Concentric Circle Datum objects which are D3-specific representations of the social media comments. These objects are used by D3's engine and the application's very own interaction event handlers to keep track and represent the state of every comment in the context of the canvas.

The procedural algorithm to convert a SocialComment into a CircleDatum is given below. A method *calculateRadius* is predefined to derive a normalized radius according to the device's screen dimensions and the min and max like count for all the comments. Additionally, a method *generateColor* is also predefined to generate a unique color for this circle based on the author's name. The replies to the comment inside the *comments* field are also recursively mapped into nested Circle Datum objects.

Input: A Social Comment *C*, radius offset *R*, pivot flag *P* **Output:** A new Circle Datum object *Result* **Procedure:** *buildCircleDatum(C, R, P)*	
1	**let** *Result* be an empty object *{}*;
2	**let** *r* = *calculateRadius(C.likeCount)* + *R*;
3	*Result.circleId* = *S.id*;

```
4   Result.color = generateColor(C.authorName);
5   Result.children = [];
6   Result.data = C;
7   Result.timestampCue = C.publishTimestamp;
8   Result.radius = r;
9   Result.isPivot = P;
10  Result.innerId = Null;
11  Result.isDisplayed = False;

12  # used in other places and by D3's simulation engine
13  Result.x = Null;
14  Result.y = Null;
15  Result.fx = Null;
16  Result.fy = Null;
17  Result.concentricCircleId = Null;
18  Result.isDisplayed = False;
19  Result.isFocused = False;
20  Result.isHighlighted = False;

21  if P is True then
22      Result.id = "pivot_" + C.id;
23  else
24      Result.id = "circle_" + C.id;
25  end if

26  let innerId = Result.id;
27  for each comment Reply in C.comments do
28      let Child = buildCircleDatum(Reply, r, False);
29      Child.innerId = innerId;
30      innerId = Child.id;
31      r = Child.radius;
32      insert Child into Result.children;
33  end for

34  return Result;
```

Algorithm 1. Build Circle Datum

The top level comments of the post will be converted into the pivotal Circle Datum objects i.e. the initial circles to be drawn at the center of every Nucleus. Throughout the Canvas Component, these special Circle Datums are referred to as Pivots. For every Pivot built by the *buildCircleDatum* procedure, a Concentric Circle Datum object is created to represent the larger element grouping which holds the set of

stacked Circle Datum elements. It is this D3 object used by the simulation to determine where on the canvas the Concentric Circle should be positioned. Among other things, this Concentric Circle object also handles selection via click events and also facilitates focusing when a keyword or similarity search is ongoing.

Input: A pivot Circle Datum *P*, nucleus flag *N*, radius offset *R*, root Concentric Circle ID *ROOT*, and parent Concentric Circle ID *PARENT*
Output: A new Concentric Circle Datum object *Result*
Procedure: *buildConcentricCircleDatum(P, N, R, ROOT, PARENT)*

```
1   let Result be an empty object {};

2   Result.id = "concentric-circle_" + P.circleId;
3   Result.circleId = P.circleId;
4   Result.peripherals = [];
5   Result.radius = P.radius;
6   Result.pivot = P;
7   Result.isNucleus = N;
8   Result.parentId = PARENT;

9   # used in other places and by D3's simulation engine
10  Result.x = Null;
11  Result.y = Null;
12  Result.fx = Null;
13  Result.fy = Null;
14  Result.isSelected = False;
15  Result.isFocused = False;
16  Result.isDisplayed = False;

17  Pivot.concentricCircleId = Result.id;
18
19  if (ROOT is Null) then
20      Result.rootId = Result.id;
21  else
22      Result.rootId = ROOT;
23  end if

24  let r = P.radius * -1;
25  let innerId = P.id;

26  for each circle Child in P.children do
27      Child.concentricCircleId = Result.id;
28      Child.radius += R;
29      Child.innerId = innerId;
```

```
30        innerId = Child.id;

31        if Child.children is not empty do
32            let ChildPivot = shallow copy of Child;
33            ChildPivot.isPivot = True;
34            ChildPivot.id = "pivot_" + Child.circleId;
35            ChildPivot.radius += r;

36            let ChildPeripheral = buildConcentricCircleDatum(ChildPivot,
      False, r + R, ROOT, Result.id);
37            Child.children = ChildPeripheral.pivot.children;
38            insert ChildPeripheral into Result.peripherals;
39        end if

40        Result.radius = Child.radius;
41        r = Child.radius * -1;
42    end for

43    return Result;
```

Algorithm 2. Build Concentric Circle Datum

To illustrate how this Concentric Circle object can be used, a snippet of the fast-forward and rewind algorithms performed when the *progress* value is changed is given below. The *newIndex* value below is the index derived from the newly updated *progress* value given by the Timeline Controls Component. The *lastIndex* value is the last seen value of this *progress* variable. The method *getCircleByIndex* is a utility routine to get a reference to a Circle Datum according to its chronological order.

```
1   if newIndex > lastIndex do

2       while lastIndex < newIndex do
3           lastIndex += 1;
4           let circleDatum = getCircleByIndex(lastIndex);
5           circleDatum.isDisplayed = True;
6       end while

7   else
8       while lastIndex > newIndex do
9           let circleDatum = getCircleByIndex(lastIndex);
10          circleDatum.isDisplayed = False;
11          lastIndex -= 1;
```

```
12        end while
13  end if

14  redraw();
```

Algorithm 3. Progress Update Event Handler

At the end of the above snippet, the *redraw* function is called which performs the actual manipulation of DOM elements present on the screen. In particular, it iterates over the list of Concentric Circles and checks the flags if any need visual changes such as changing of opacity levels or whether to show the boundary box element or not.

5.2.2 Force-Directed Simulation Configuration

The layout design built using D3 JS library makes use of 6 different forces in a single force-directed simulation namely: Many Body Force which acts as the repelling agent that pushes peripheral concentric circles outwards the nucleic concentric circle, a Point Force X to act as the gravity force horizontally attracting the bodies towards the center of the canvas, a Point Force Y to act as the gravity force vertically attracting the bodies to the center of the canvas, Immediate Relation Collision Force which controls the collision between immediately related concentric circles, Relation Collision Force which controls the collision between concentric circles part of the same tree and a general Collision Force which is the default handler for non-related concentric circles.

Independently running on a separate simulation is another force named Link Force which is responsible for the *orbit* effect that makes the peripheral concentric circles surround the parent concentric circle. The separation of this force from the primary simulation is to avoid irregularities between conflicting forces as we would need to make the peripheral concentric circles act as if they were independently attached to the parent concentric circle. Doing so will decrease the number of bodies to be

calculated for the heavier simulation and also allow the peripheral concentric circles to flow naturally without influence from the other forces.

The Many Body Force allows the simulation to enact an electrostatic charge (repulsion) between the concentric circle bodies belonging to the same social discussion tree but are not immediately related to each other. By applying this force, arrangement of peripherals around a nucleus is more evenly spread, avoiding awkward fat-thumb issues when the user taps on a peripheral concentric circle. The strength of this force is a function of the concentric circle's outermost radius multiplied by a negative (repulsive) coefficient scaled to the device's screen size.

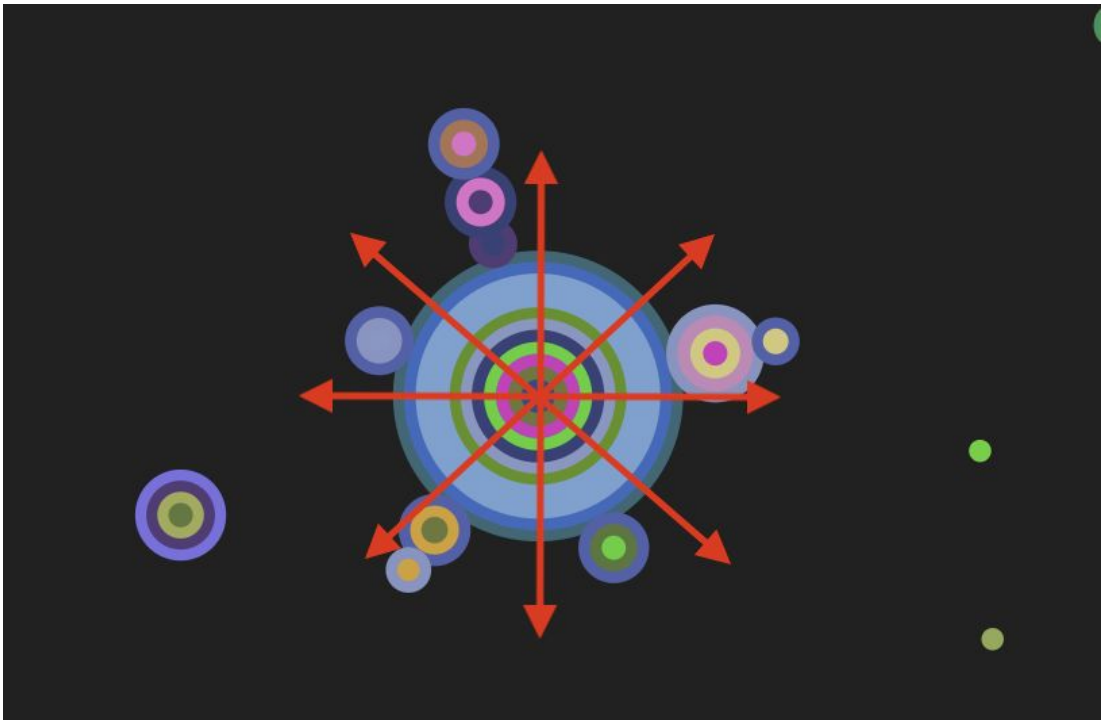


Figure 21: Many Body Force

The Point Force X and Point Force Y both enact an artificial gravity which pulls all concentric circle bodies towards the center of the canvas. This is the same force responsible for the conversation landscape's circular shape.

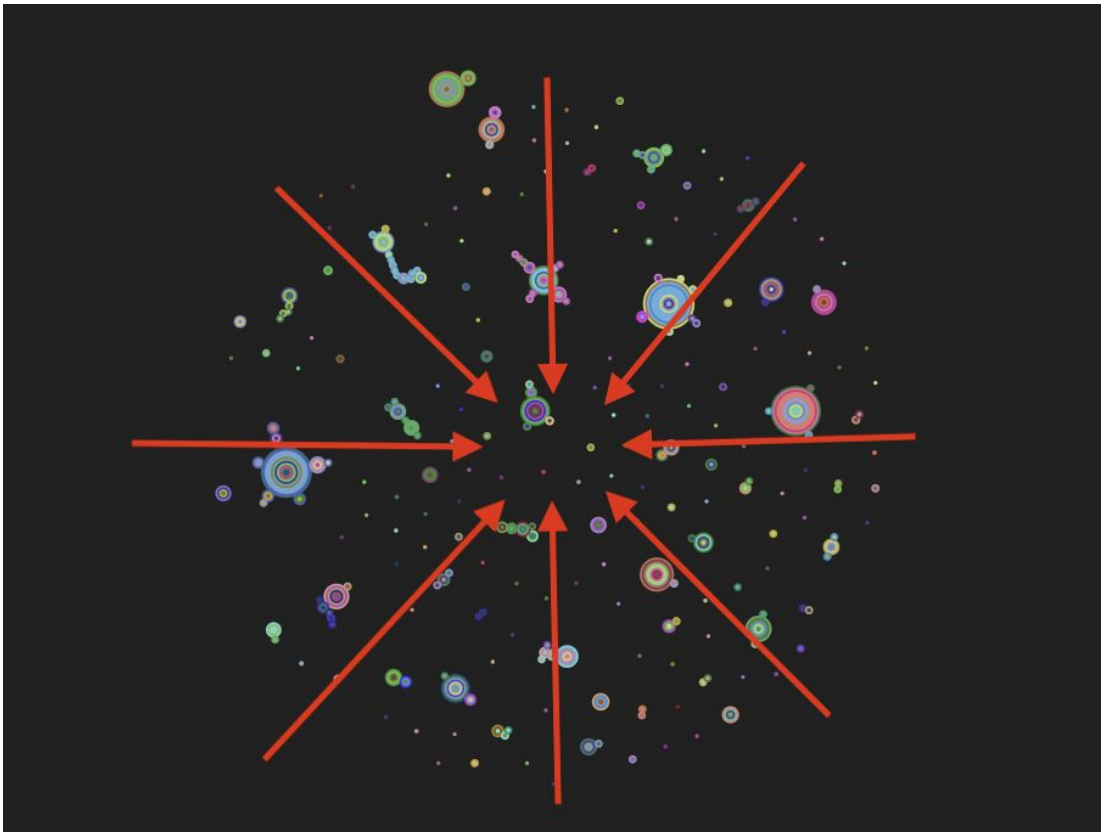


Figure 22: Point Force X and Point Force Y

The Immediate Relation Collision Force enforces a slightly overlapping collision rule between two concentric circle bodies that belong to the same discussion tree and that are immediately related to each other. The collision radius is set as the concentric circle's radius minus a small constant to give the overlapping effect that appears between immediately related concentric circles.

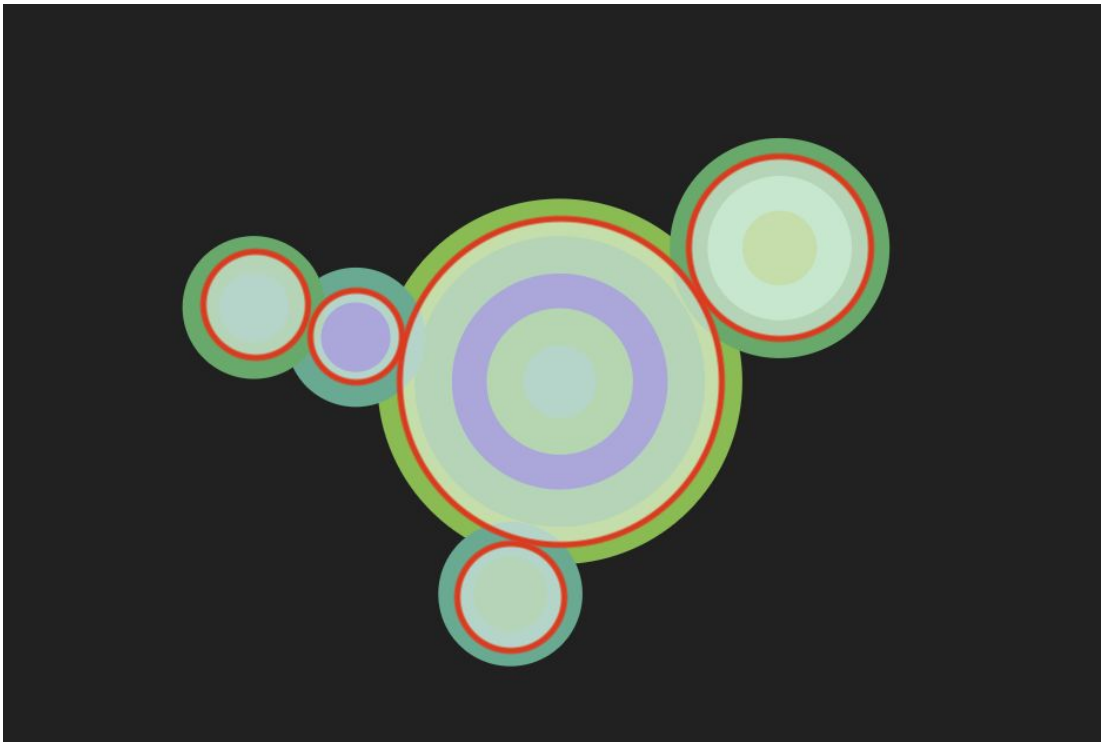


Figure 23: Immediate Relation Collision Force

The Relation Collision Force is another collision rule which handles the non-overlapping collision rule between two concentric circle bodies that belong to the same discussion tree but are not immediately related to each other. This force primarily ensures that there are no overlaps between peripheral concentric circles belonging to the same root nucleic concentric circle. Unlike the Immediate Relation Collision Force, this force implements zero overlap between the concentric circle bodies.

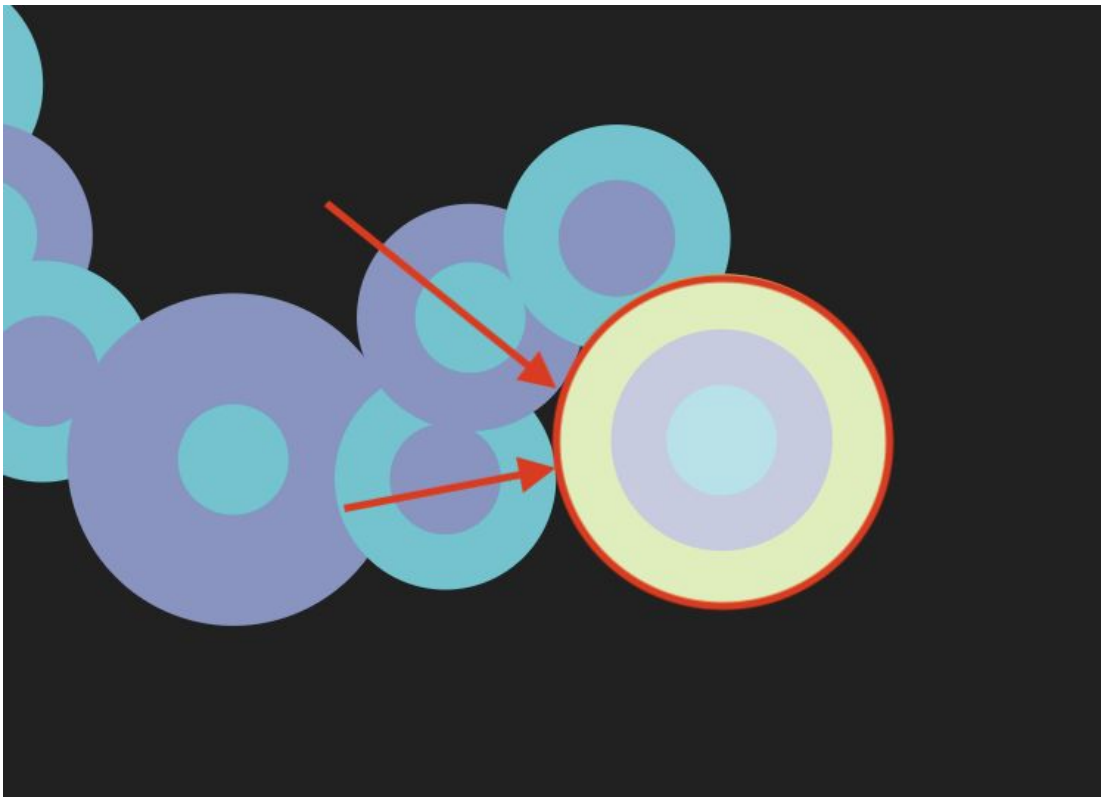


Figure 24: Relation Collision Force

The General Collision Force is the last kind of collision rule in effect which handles the general non-overlapping collision rule between two concentric circle bodies that do not belong to the same discussion tree. This force also applies the minimum spacing required between two concentric circle bodies by adding a positive constant to the concentric circle's radius resulting in an invisible extension to its actual radius.

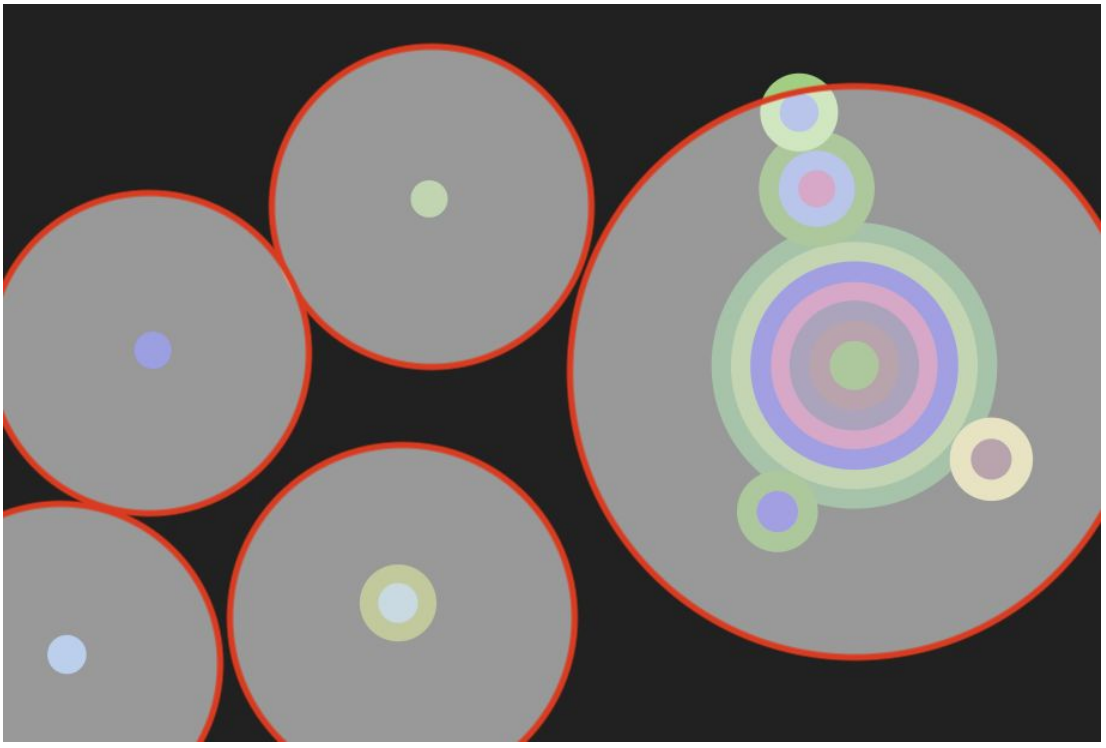


Figure 25: General Collision Force

Lastly, the Link Force applied on a separate simulation facilitates the connectedness of peripheral concentric circles to their parent concentric circle. This is a modification of the generic Force Simulation's default Link Force since this overwrites the distance between related concentric circles as the larger radius between two concentric circles instead of using a fixed constant. By dynamically setting the distance between two linked concentric circle bodies, we achieve the orbit effect, making the peripherals position themselves around their parent concentric circles.

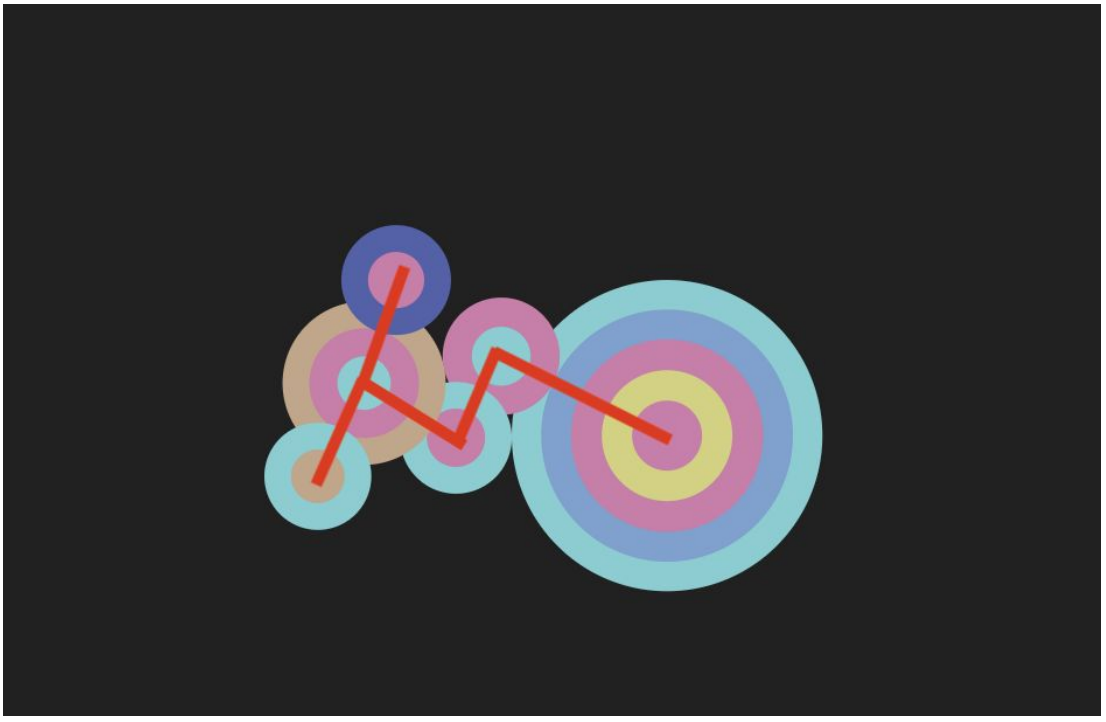


Figure 26: Link Force

5.3 Comparison with Baseline

KANDINSKY's custom circles layout is compared with three baselines - the existing mobile view, the generic force-directed simulation layout and the traditional hierarchical network view, based on the seven criteria proposed by Few and Edge (2017) to evaluate the data visualization effectiveness - usefulness, completeness, perceptibility, truthfulness, intuitiveness, aesthetics and engagement.

Usefulness is determined in light of the user's needs which is to effectively understand and analyze the social discussions such that important insights can be generated immediately and effortlessly.

Completeness refers to providing information that is required to produce the intended level of understanding, neither more nor less. This involves the right (i.e. relevant) information and the right amount if it.

Perceptibility means the information must be displayed in a manner that the user can perceive with minimal effort.

Truthfulness refers to the accuracy and validity of the displayed information. Accuracy measures reliability and precision, while validity indicates appropriate representation.

Intuitiveness indicates whether the display is familiar and easily understood.

Aesthetics measures whether the display is pleasing to the eye such that the user intends to explore out of their own natural interests.

Engagement is a quality achieved when users are drawn into the data exploration.

Table 11 summarizes the comparison result.

Criteria	KANDINSKY	Existing Mobile View	Forced-Directed Simulation	Network View
Usefulness	High	Low	Medium	Medium
Completeness	High	High	Medium	Low
Perceptibility	High	Low	Medium	High
Truthfulness	High	High	High	High
Intuitiveness	High	Low	Medium	Medium
Aesthetics	High	Low	Medium	Medium
Engagement	High	Medium	Medium	Medium

Table 11: Comparison of KANDINSKY's Circles Layout with Baselines

5.3.1 Existing Mobile View

For the existing mobile view, all comments are represented to the user in the strict format of listed text elements. The hierarchy of threads made flat, such that even social discussions deviating from the thread are grouped under in the same level under the thread.

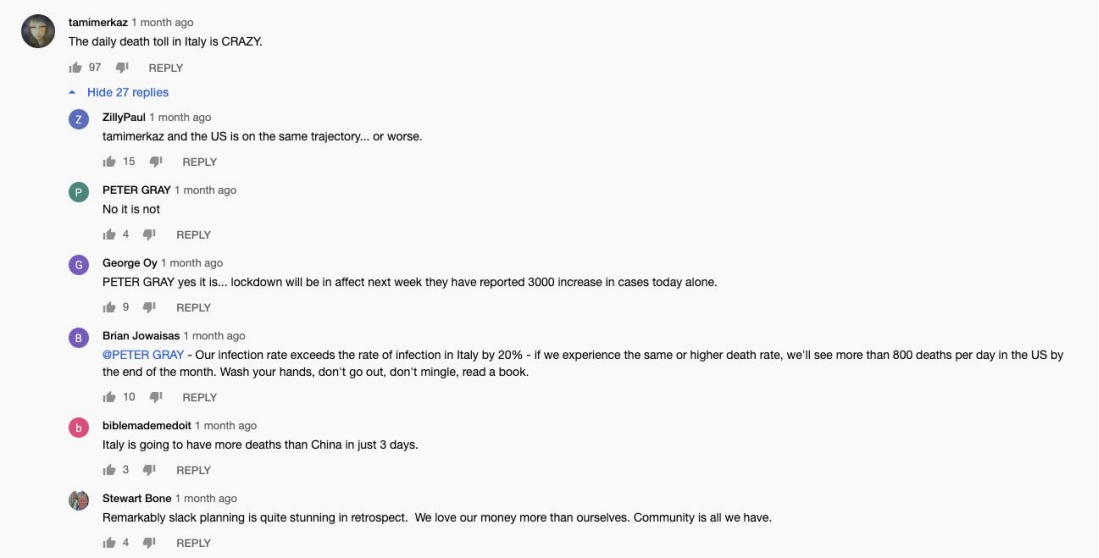


Figure 27: Existing Mobile View

Text data is certainly important, but raw text provides overwhelming detailed information without the ability to look at a certain comment with reference to the social discussion landscape. This results in a limited usefulness for the user to understand and analyse comments. Although this view provides informative completeness, the perceptibility factor suffers since there is a lot of effort required to do some level of analysis. Truthfulness is fulfilled since the original data is displayed. However, aesthetically, pure text is not pleasing to the eye, especially given the huge volume of text. As a result, nothing invites the user to examine and explore the data, leading to little user engagement.

5.3.2 Force-Directed Simulation Layout

With the generic force-directed simulation layout, all comment threads are positioned arbitrarily as floating trees of connected nodes on a canvas along with edges between related nodes to denote the continuity relationship between them. The center of the graph would be an abstract node representative of the post itself. Unlike the existing mobile view, the hierarchy of threads are not made flat, such that social discussions deviating from the threads are displayed as outgoing paths to more connected nodes.

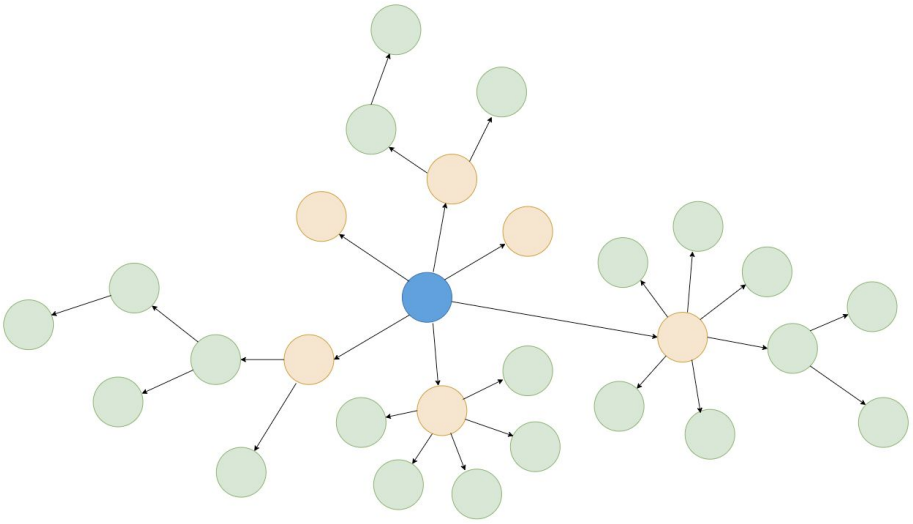


Figure 28: Generic Force-Directed Simulation Layout Concept

Although nodes can be sized dynamically according to the comment's number of likes, this method would occupy more space since this network-like view would require every comment to be represented as a separate node.

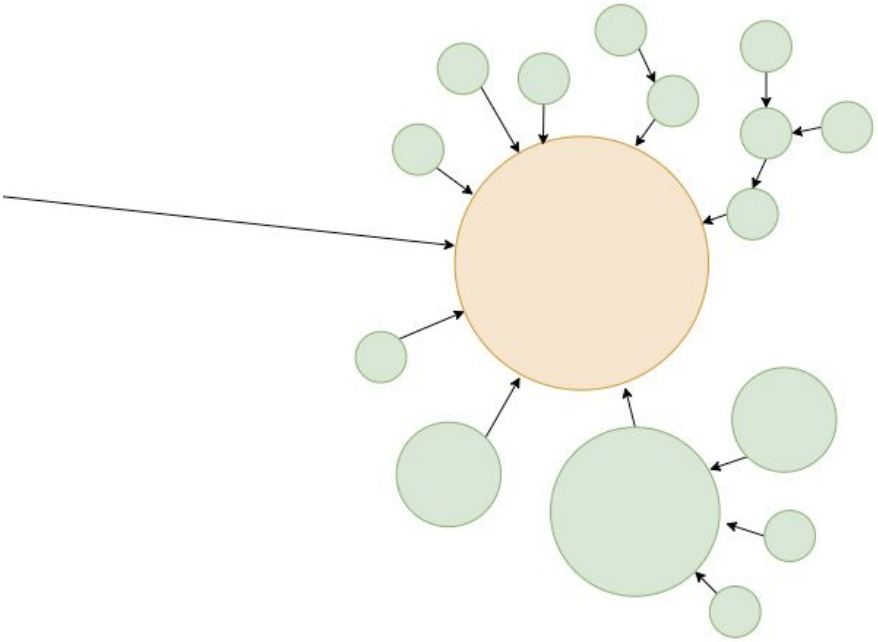


Figure 29: Dynamically Sized Circles in Force-Directed Simulation Layout

That said, edges provide good visualisation to intuitively represent commentary hierarchy, it may not be quick for the user to perceive the direction of the conversation since doing so would require some symbol to denote the direction of the edges. In terms of its usefulness, the weight of the edges could be used to represent the time difference between two commentary nodes, but this would result in a very irregular or imbalanced shape since there is no enforceable time boundary that limits this value. This makes edges redundant or even useless to some extent as it simply occupies more space without providing any additional benefit. Alternatively, keeping the edges and setting the weight as a fixed value can make the layout more inviting and engaging. However, the polluting and overlapping lines between nodes would be an eyesore to the user, degrading the layout's aesthetic factor.

5.3.3 Network View

The traditional top-down network view is very similar to the force-directed simulation layout described in the previous section. However, nodes are placed in a top-down pattern, where the head of the network tree is the post and the nodes branching dowards are the replies to it. Although this time, the direction of conversation continuity can be perceived more clearly without the need for arrows, the edges would still require the layout to occupy a significant amount of space for such little informational value.

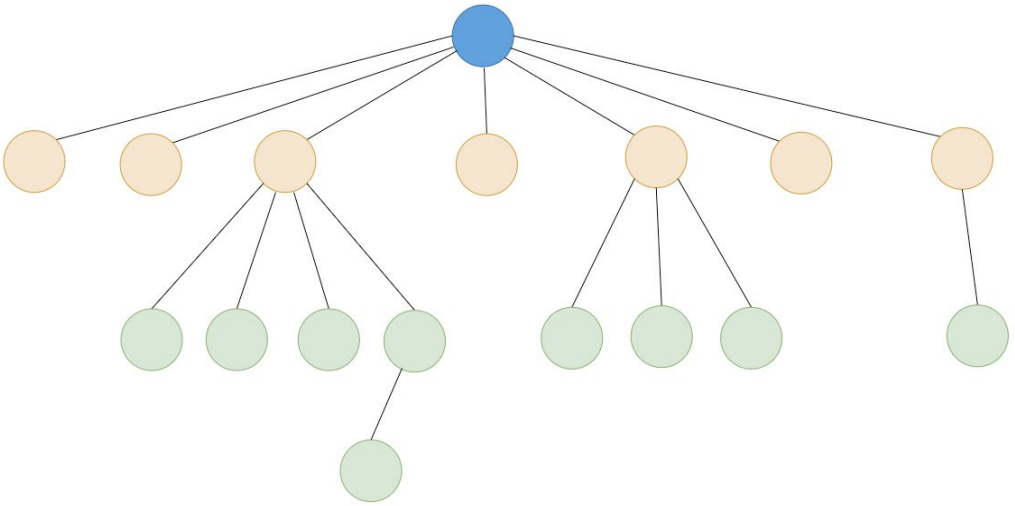


Figure 30: Traditional Network View Concept

With this layout, a drawback could be the direction of canvas expansion being biased to grow vertically or horizontally. For posts that have a lot of top-level comments, the network view would quickly expand horizontally. For posts that have long-winded conversations, the network view would quickly expand vertically. Both cases would require the user extra effort to scroll the canvas sideways or vertically depending on the network's shape. The same disadvantage results in a lower completeness since the user would have a more difficult time comparing clusters of comments made earlier (top of the tree) vs clusters of comments made later (bottom of the tree).

5.3.4 Circles Layout

Though only overview information is provided initially, detailed information and relevant analysis can be accessed when needed, achieving informative completeness. The layout is also very useful in helping the user understand the social discussion landscape and identify interesting threads easily.

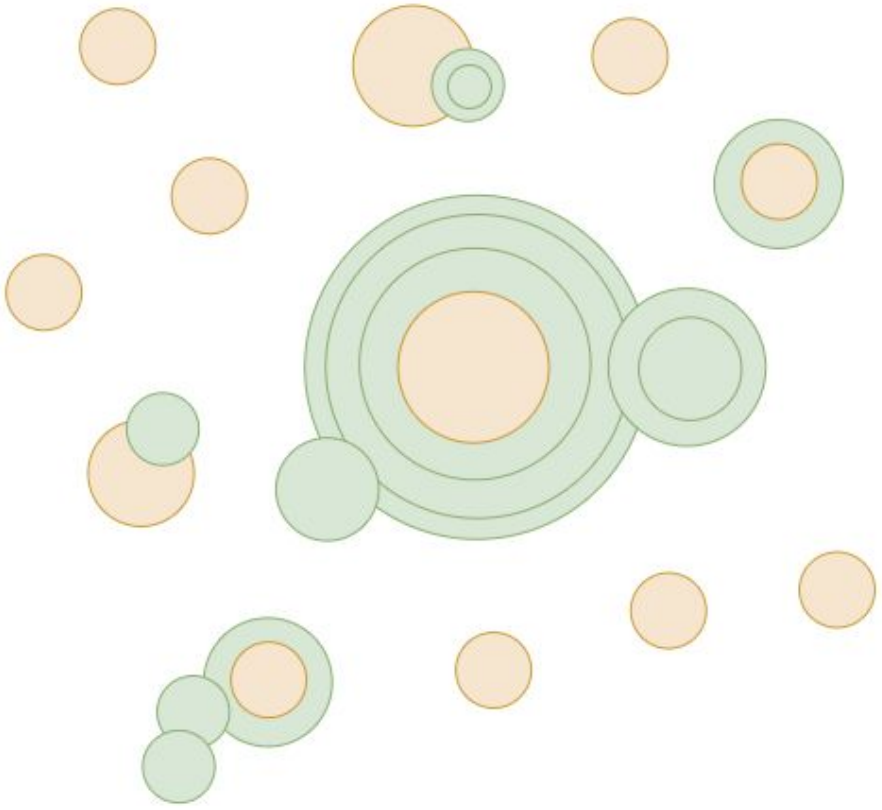


Figure 31: Circles Layout Concept

The well designed layout achieves a high level of perceptibility and intuitiveness, as the circles layout allows clear and easy understanding of social discussion landscapes. Truthfulness is fulfilled with accurate information representation of each comment by the concentric circles. Aesthetically, it is inspired from the famous painting, thus being most visually pleasing. Being easily understood and aesthetically pleasing also help encourage the user to explore and analyse the information, achieving user engagement.

6. Search & Analytics Module

This section discusses in detail the modules involved in the search and analytics processes of the application.

6.1 Evolutionary View

The evolutionary view of the application is embodied by the Canvas Component and the Timeline Controls Component which are both responsible for controlling the visualisation of commentary concentric circles based on a tracked timestamp value called *progress*. This *progress* variable is an incremental index counter controlled by the Timeline Controls Component which points to the next comment to be displayed according to its chronological order, starting from -1. A value of -1 would refer to the state of the post when it was just newly published and no comments had been submitted yet. This mechanism would also imply that the max value of *progress* would be the total number of comments minus 1 and when at this state, all comments available would have already been displayed in the canvas. Both values are set when the Timeline Controls Component's slider is at the opposite ends of the range slider.

When the application is in play mode, the value of *progress* is incremented by a value of 1 for every 100ms. This iterative routine is done in the Timeline Controls Component itself and propagates the changes to the value of progress to other components that rely on it, like the Canvas Component and the Kandinsky Interface Page. The Canvas Component would use this updated value to identify which circle to visualize next and the Kandinsky Interface Page would use this updated value to update the current time displayed in the interface. Every change to the value of progress is automatically propagated in a stateless manner. This design allows the user to use the range slider to *jump* across points in time like he/she would on a video player's seeker bar.

On the Canvas Component, every circle's DOM element is already built and positioned in the initialisation phase of the canvas. To emulate the evolutionary

effect, every circle is hidden by setting their opacity values to 0. When the *progress* value is updated, this component would match the new value by changing the opacity values of the eligible commentary circles to 1 or 0 depending on whether the new value of *progress* is more than or less than the last seen value. As part of the routine which controls a circle's visibility, subroutines are also executed to resize the circle's click box boundary, update the eligibility of the circle to be tapped by the user, and the eligibility of the circle to be a candidate search result when a search is performed. Internally, when the new value of *progress* is greater than the last seen value, it is referred to as a *fast forward* update. Conversely, when the progress is less than the last seen value, it is referred to as a *rewind* update. Update events that arise when both values are equal are ignored since there would be no change in the state of the canvas.

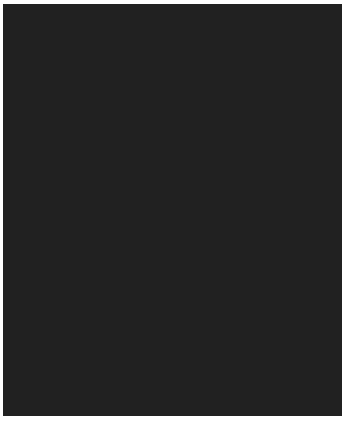
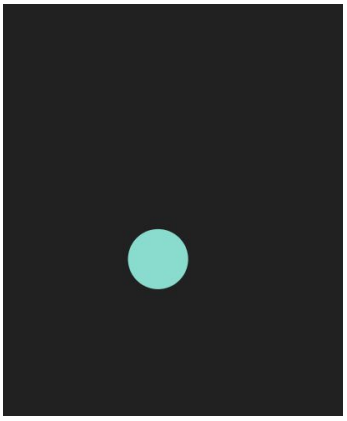
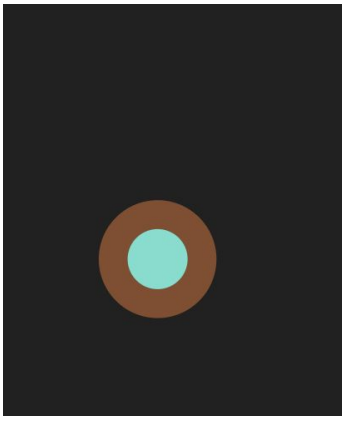
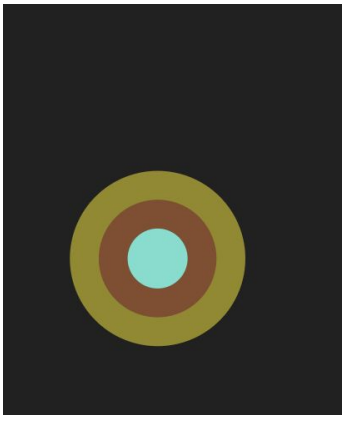
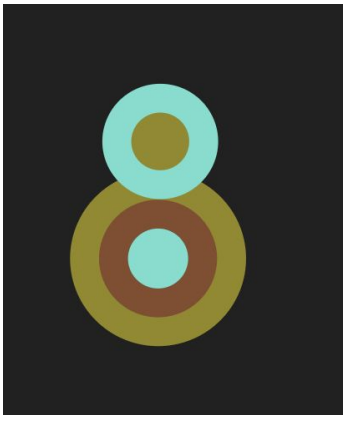
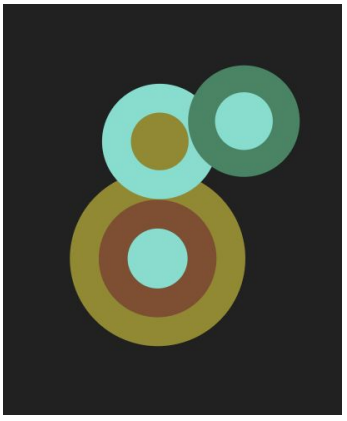
 **Progress = -1**	 **Progress = 0**	 **Progress = 1**
 **Progress = 2**	 **Progress = 3**	 **Progress = 4**

Table 12: Incremental Fast-forward Update

6.2 Similarity Search

A bulk of the requirements to do similarity search function is done in the preprocessing phase of the extraction process as described in section 4.3 Data Preprocessing. When the user performs a similarity search, the application will use the analytics metadata generated by the preprocessing phase to identify which comments to be focused. This eager approach to searching for similar comments results in faster turn-around time as compared to when doing it lazily on demand.

The similarity search is triggered when the user clicks on the "view similar comments" text inside a comment item within the comments list. This prompt is only made visible for comments which have been identified to have similar topic distributions with other comment items. When this prompt has been clicked, the current comment would be set as the reference comment and it would be affixed at the top part of the Kandinsky Interface Page. The list of comments below would also be temporarily replaced by the identified similar comments of the reference comment set. When this happens, the same set of similar comment IDs would be passed to the canvas component for focusing.

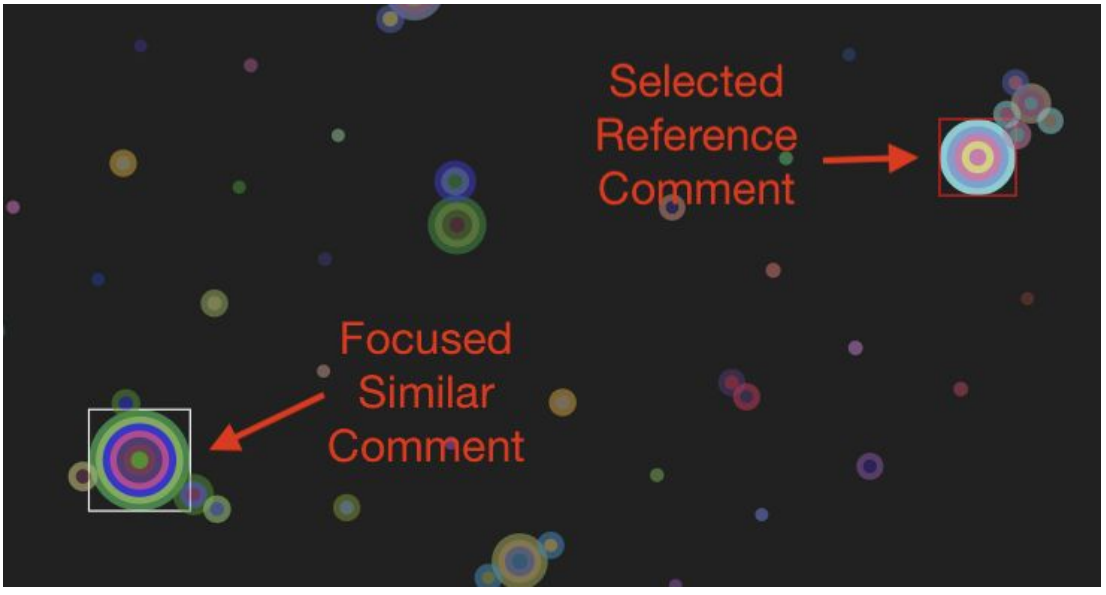


Figure 32: Focused Similar Commentary Circles

In the Canvas Component, when a set of circles is being focused, all currently visible circles will have their opacity values halved except for the set of focused circles. To improve the perceptibility of this effect, white bordered boxes will be displayed around the focused circles to make them easier to stand out to the user.

6.3 Keyword Search

When a keyword search is performed in Kandinsky Interface Page, the *search* method passes the query string to Kandinsky Service's *searchComments* routine which filters out the active post's comments that match the query. A Regular Expression pattern is used to perform the search algorithm which returns the matching comments along with their respective indices of the query string's position on the textual content. The *searchComments* function returns a list of Search Result objects which is described by the schema given below.

```
{
  commentId: string;
  indices: [number, number][];
}
```

Table 13: Search Result Schema

When the Kandinsky Interface Page receives this, it passes the set of matching comment IDs to the Canvas Component for focusing. The focus effect is exactly the same routine called when a similarity search is performed as described in the previous section 6.2 Similarity Search. This focus effect will remain until the search text box is cleared or until the user performs focus-overriding functions such as similarity search.

While the keyword search is active, clicking on a matching circle will display its comment thread and will make use of the search result's indices list to highlight in blue color the search query's position within the comment's textual content.

6.4 Spectrum

The spectrum bin generation is a critical step to prepare the data to be visualised for the Spectrum Component. Since the life duration of UI components tend to be short, the generation of spectrum bins are done in the Kandinsky Service when the canvas is initialised where the bins can be retained and memoised along with the set active post. By doing this, we avoid recalculating the bins every time we open the Spectrum Component.

Within the service, there is a method named *groupCommentsByTimestamp* which receives a numeric argument called *groups* set to 100 by default. This value will be used to determine how many bins to group the active posts' comments to. From the date the post was published to the date of the last comment published, a UTC scale is built which can be used to inversely calculate a timestamp value given an index from 0 to *groups - 1*. For example, for a grouping of comments into 100 bins from a video posted on 01 January 2020 1200 with the last comment published on 28 April 2020 1200, an index value of 0 would be mapped to the UNIX timestamp value of 1577880000 representing the timestamp for the video's date of upload. Using the same example, an index value of 99 would be mapped to the UNIX timestamp value of 1588075200 representing the timestamp of the last comment published. With a time range of 118 days apart, each bin would represent an interval of 28.32 hours each.

This list of objects representing each bin is created, each holding the start and stop timestamps of the interval, an empty list to store the comments that fit in the interval and a count initialised as 0 to count the number of comments in the said list. Following that, the method iterates over the active posts' comments sorted by timestamp and uses a pointer variable to keep track of which current bin we're in, starting with bin 0. Comments that fit the current bin are added to it. If a comment's publish timestamp is found to be greater than the current bin's stop timestamp, the

pointer is incremented until the appropriate bin with overlapping time interval is located. The model representing each bin is described below.

```
{
  start: number,
  stop: number,
  count: 0,
  comments: SocialComment[]
}
```

Table 14: Comment Group Interval Schema

Once all bins are prepared, the list of Comment Group Intervals can be used by the Spectrum Controls Component whenever the user activates it. In the Spectrum Controls Component, the bars are drawn using the bins generated. The *count* field of the Comment Group Interval Schema would be mapped to a logarithmic scale value to be used as the bar's height on the spectrum canvas. To allow the Spectrum Controls Component to communicate with the Canvas Component, each bin's *count* field is also summated accumutavely in order to allow it to translate as a *progress* value. When propagating changes in the Spectrum Control Component's slider knobs, *lower* and *upper* progress values which represent the currently highlighted range of bars are passed to the parent Kandinsky Interface Page. Using the same list of Comment Group Intervals, the pair of *lower* and *upper* progress values are both reduced into the list of comments that fit within the range interval.

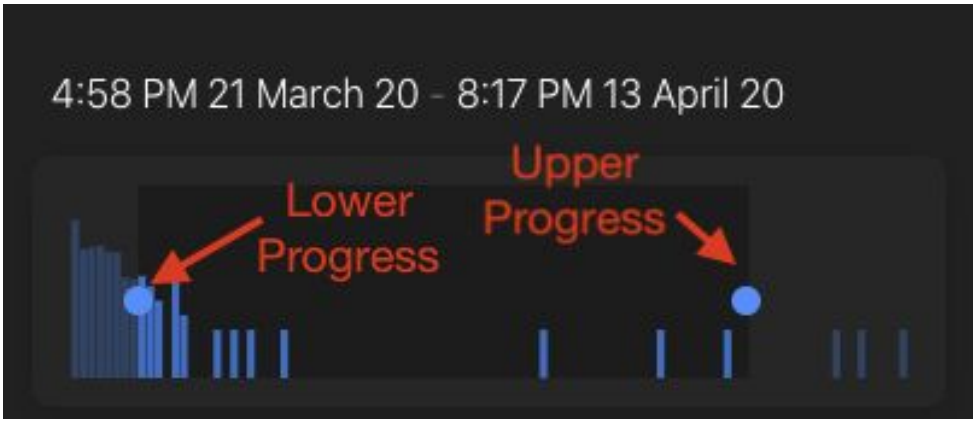


Figure 33: Spectrum Controls

The list of comment IDs derived from the steps described above is then passed to the Canvas Component for highlighting. When spectrum mode is enabled, all comments will be made visible by default, with a lower opacity to improve perceptibility of highlighting the commentary circles contained within the spectrum controls range. Highlighting differs from focusing (as done in keyword and similarity search) by not rendering a white bordered boundary box around those highlighted circles. When this spectrum mode is active, the user can proceed with searching or navigating the canvas as he/she would when the spectrum mode is inactive.

7. Visual Interface

This section demonstrates the UI design principle and implementation with a walkthrough. A detailed demonstration video can be found online[1].

7.1 Design Principles

7.1.1 Information Design Hierarchy

As mentioned in the literature review section, Hoober (2017) states that people's touch interactions are faster, more confident, and more accurate as they approach the center of the screen. Based on this observation, he proposes this hierarchy of information design as summarized in Figure 34.

Applying this framework, comments with most replies are the most important content, thus being placed at the center of the canvas layout.

[1] https://youtu.be/lGwEwz9qyMc

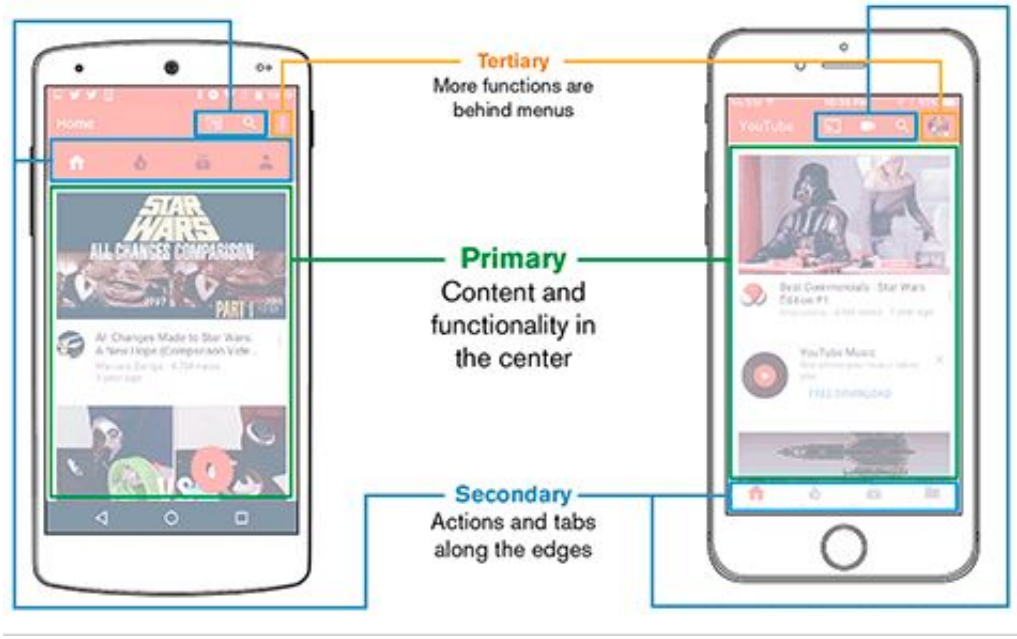


Figure 34: Touch-friendly information-design framework

7.1.2 Color Usage

One design consideration related to color usage is the choice of dark background color. The choice of a darker shade of gray instead of pure black prevents eye strain due to overstimulation of the ON ganglion cells in the eyes.

In addition, studies recommend the use of contrasting color boundaries instead of line boundaries since it helps the shapes stand out and enhance absorption of information from screens. Thus, circles are differentiated through contrasting colors instead of line boundaries.

7.2 Various Components/Features

7.2.1 Post Menu Selection Page

The post menu selection page (Figure 35) allows the user to set the post to be opened. The user can choose to either open an existing visualized post or to add a new post by submitting the post's public URL. The post the user chooses is referred to as the anchor post and only comments and replies to its comments are then selected to be visualized within the app.

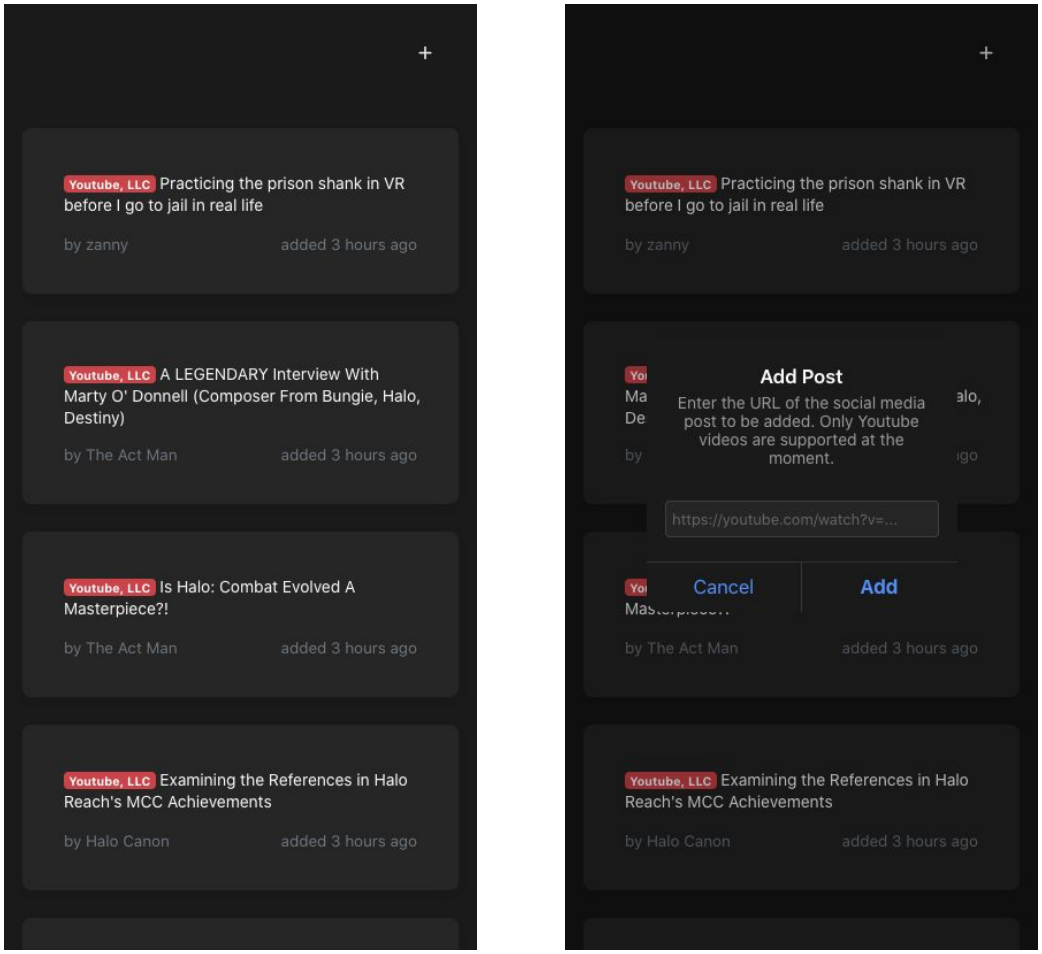


Figure 35: Post Menu Selection Page

7.2.2 Canvas Page

The canvas page (Figure 36) consists of the anchor post information card, the search bar, the circles layout, the timeline control slider and the current timestamp. The anchor post information card displays the post title and author. The search bar allows the user to enter the search keyword. The circles layout display concentric circles representing comments and their replies. The timeline control slider allows the user to visualize and control the current timestamp which dictates and filters which comments are to be displayed when the application is in Play Mode.

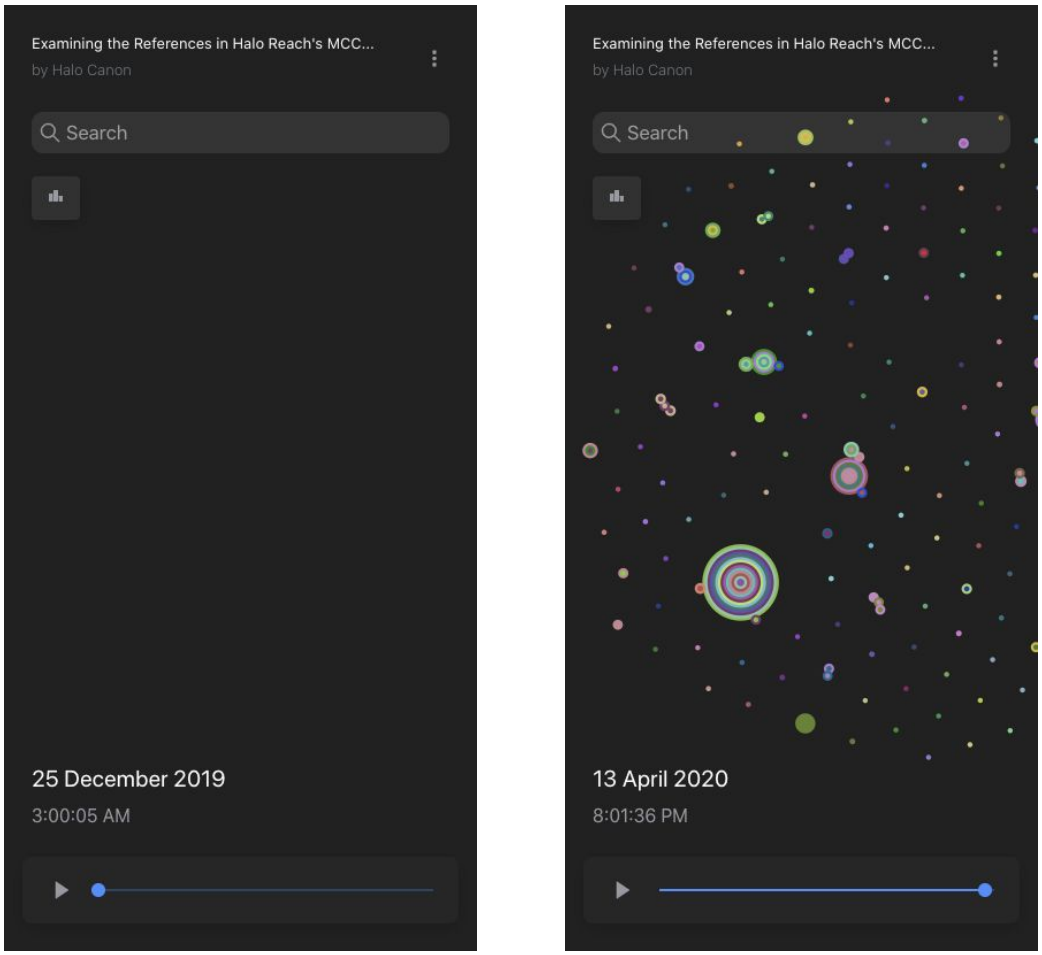


Figure 36: Canvas Page

When the application is in play mode, the application incrementally draws one circle at a time on the canvas representing the extracted comments of the anchor post sorted in chronological order. Each circle's radius is derived as a function of the number of likes the comment has. A color is assigned to each commentator and the same color is then used to paint all of the commentator's comments within the same comment thread. As more commentators join the discussion, replies to top-level comments are similarly drawn on the top-level comment's center with a radius incremented by an offset to make it slightly larger than the parent comment. As the discussion comment thread grows deeper, commentators often spawn subset discussion threads within the parent's nucleic discussion thread. These comments are commonly referred to as replies to comments and such comments may not necessarily be directed to the anchor post's content. In an attempt to retain the hierarchical relationship between comments and its replies, the peripheral concentric circles are set to be drawn within close proximity, often almost overlapping to its nucleus.

7.2.3 Detailed Information

Similar to the implementation of its desktop counterpart, KANDINSKY Mobile implements details-on-demand design where detailed information about comments can be viewed upon request. Clicking on a concentric circle will display the contents, the author names, the number of likes and the respective dates the comments were published (Figure 37).

With respect to the limited space available on mobile devices, the displayed comments list is restricted to a fixed height adjusted as a portion of the device's vertical length. The colored bars drawn on each comment item represents the comment like count normalized according to a scale derived from the discussion thread it belongs to. To enhance user navigation, comments with replies to it or comments that are replies to a parent are presented with shortcuts to allow the user to quickly select the child or parent concentric circle and also pan the canvas to the related concentric circle's location. Keeping a concentric circle selected while the

application is in play mode will automatically add or remove the comments to or from the list of selected comments.

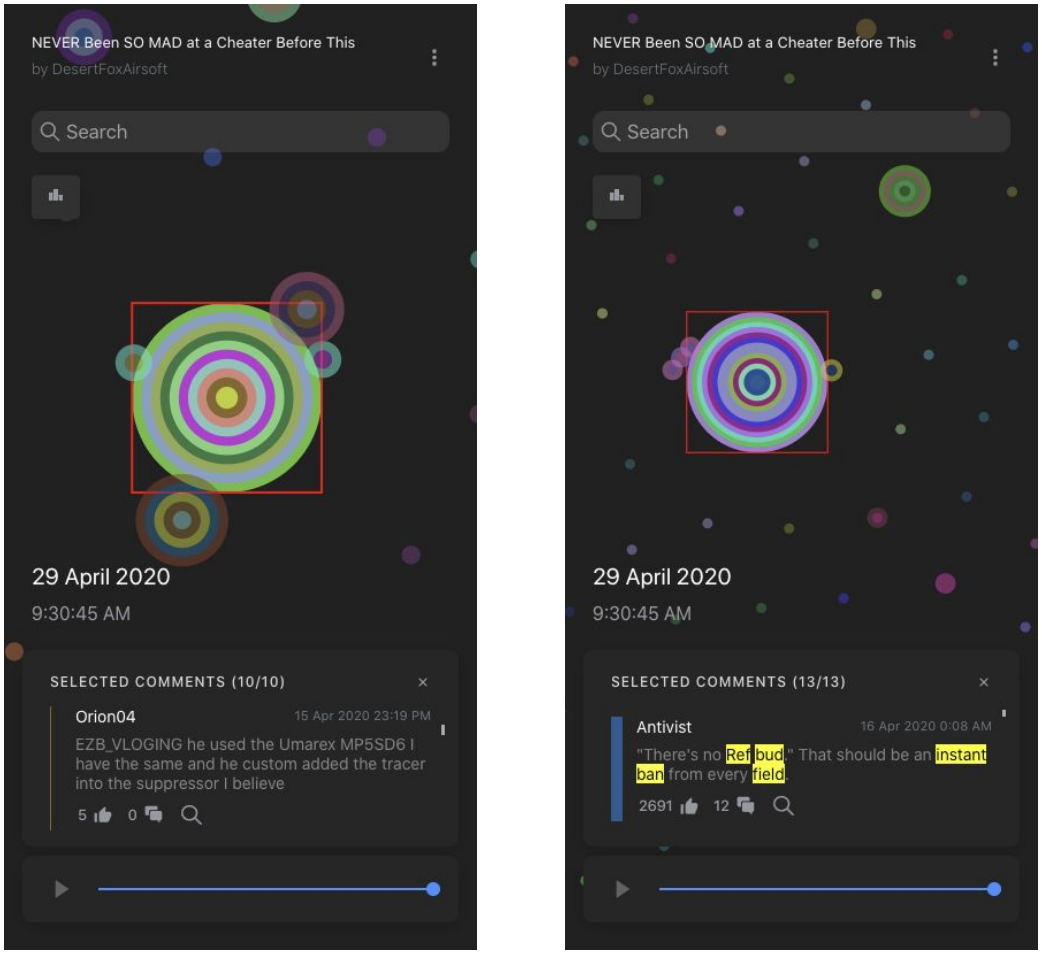


Figure 37: Detailed Information

7.2.4 Spectrum Control

The spectrum control allows the user to visualise the distribution of discussion activities over time, and control which specific comments to be highlighted given a specific time interval. When the spectrum mode is activated, all concentric circles are immediately displayed and the timeline player controls are replaced with a spectrum interface where the user is presented with a bar-chart like visualization representing the number of comments published over time (Figure 38). This gives the user insights about changes to the posts's level of discussion activity from the date the first comment was published to the date of the latest comment extracted. Using the

spectrum controls, the user is able to filter comments displayed between a specified time frame where new comments published within the time frame are highlighted. In contrast, comments that were not published within the said time frame are blurred out by setting their display to lower opacity levels.

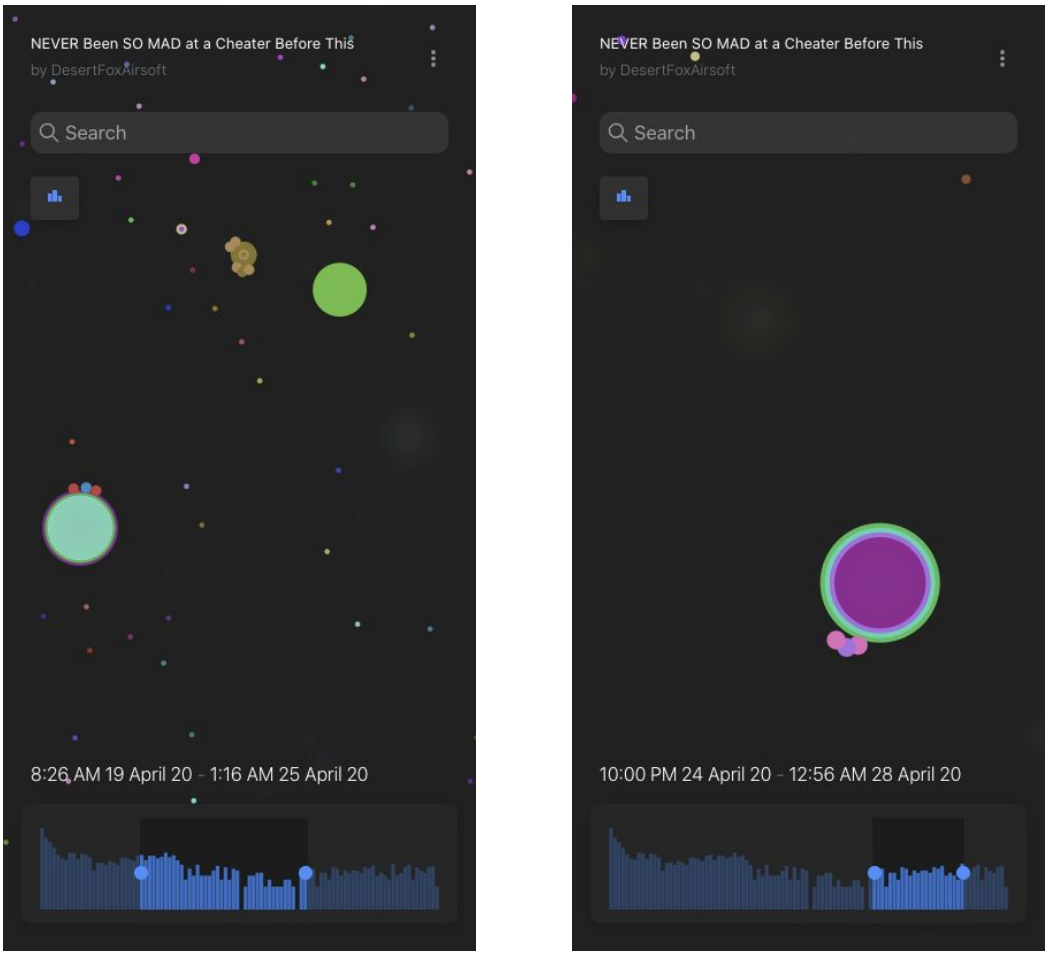


Figure 38: Spectrum Control

7.2.5 Similar Comments

KANDINSKY Mobile also implements topic modelling on the content of each comment to enable the user to see other comments with similar topic distribution (Figure 39). Identified topics within the textual content are highlighted in yellow. The reference comment is affixed at the top of the interface while the list of similar comments are listed below, just like how concentric circle comment details are

displayed when selected. Each comment item features a zoom button to allow the user to quickly pan the canvas to its circle's position.

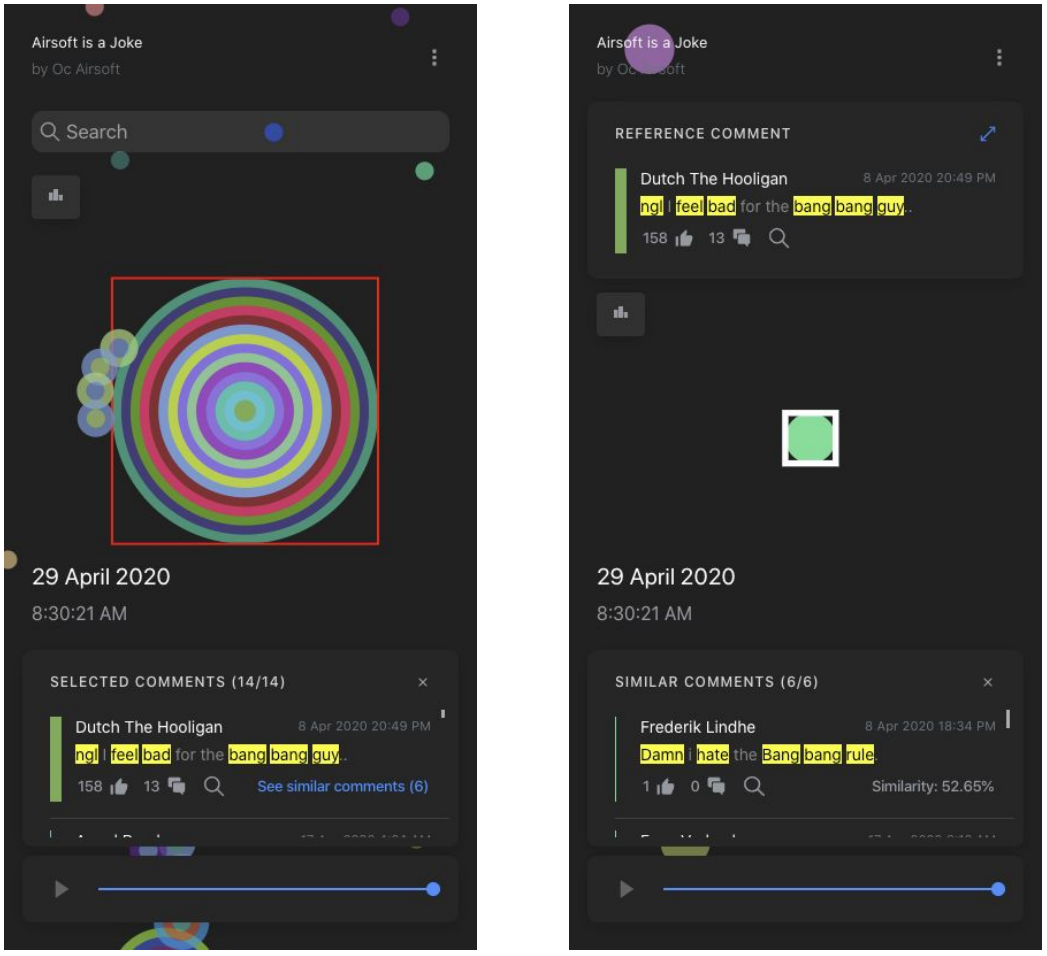


Figure 39: Similar Comments

7.2.6 Keywords Search

KANDINSKY also features a search function where the user is able to search for comments that contain a given set of keywords. Concentric circles that contain comments matching the given keywords are highlighted and surrounded with a white boundary box (Figure 40). Selecting one of the highlighted concentric circles will display its comments as expected but will also highlight the matching keywords in blue.

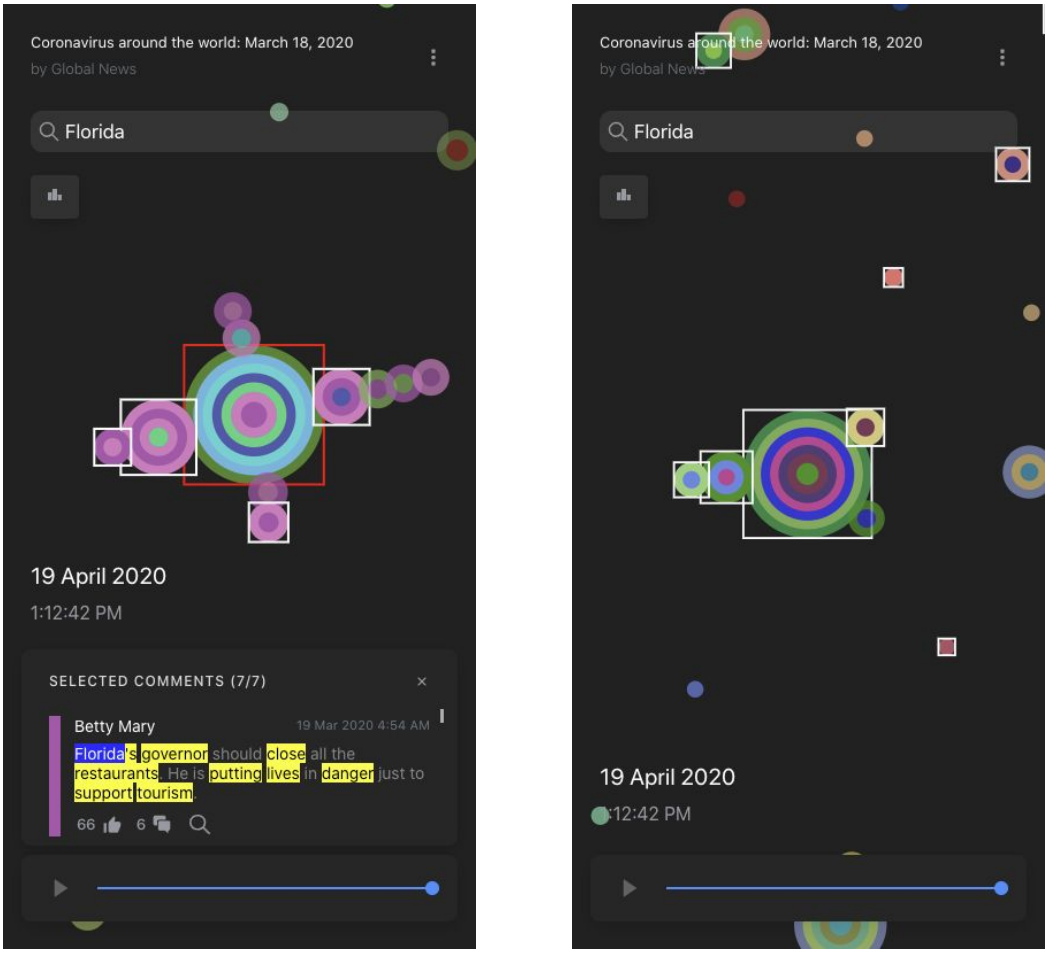


Figure 40: Keywords Search

7.3 Walkthrough

On March 18, 2020, Global News published a video on Youtube titled "Coronavirus around the world: March 18, 2020"[2], providing a roundup of the top headlines about COVID-19 coronavirus all around the world. In this section, analysis of this post is conducted using KANDINSKY Mobile.

7.3.1 Selecting Post

When the application is opened, the user is directed to this post menu selection page (Figure 41) to set the anchor post of which comments and replies to its comments are then selected to be visualized within the app. Here, the post is added by entering its

[2] https://www.youtube.com/watch?v=1pB0-nHF8Jc

public URL of the Youtube video “Coronavirus around the world: March 18, 2020”, after which, KANDINSKY first extracts the comments from the social media platform’s API sorted by the comments’ respective relevance.

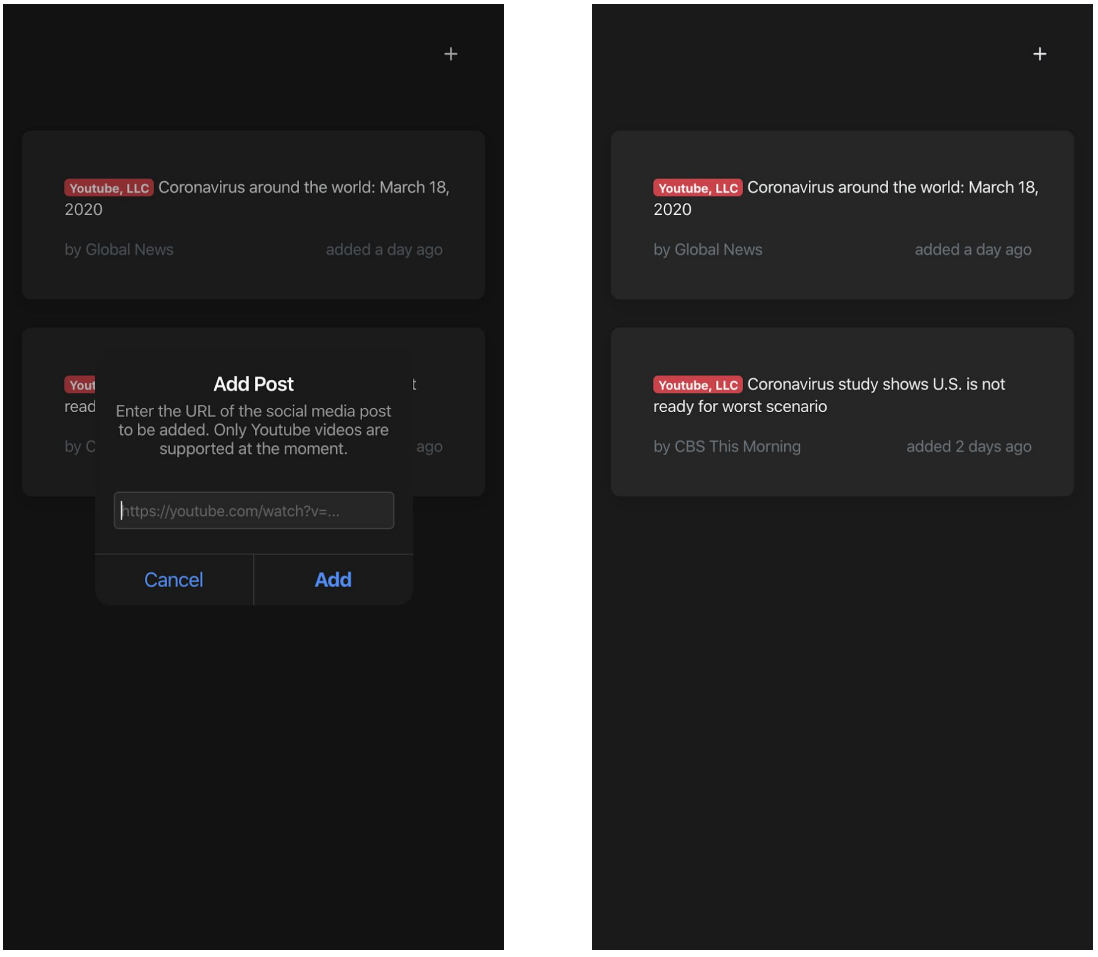


Figure 41: Selecting Post

7.3.2 Visualizing Canvas

Once the data has been extracted and prepared, the user is directed to the default canvas page where the control interface is presented (Figure 42(a)). The post title "Coronavirus around the world: March 18, 2020" and the author name "Global News" are displayed at the top area of the screen. The circles layout is empty initially as the timestamp is before the first comment being posted.

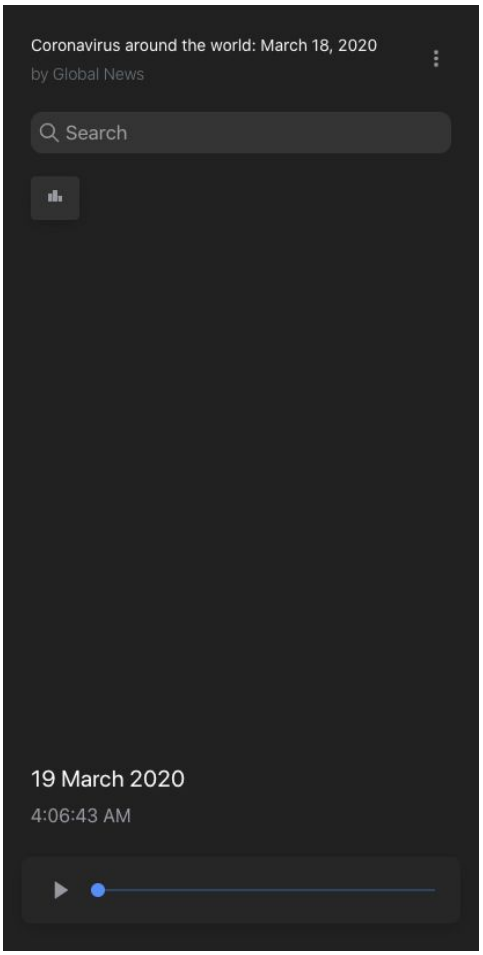


Figure 42(a): Viewing Canvas Page - Start

Clicking the start triangle button of the timeline control triggers the play mode, after which the application starts to incrementally draw one circle at a time on the canvas representing the extracted comments of the anchor post sorted in chronological order (Figure 42(b)).

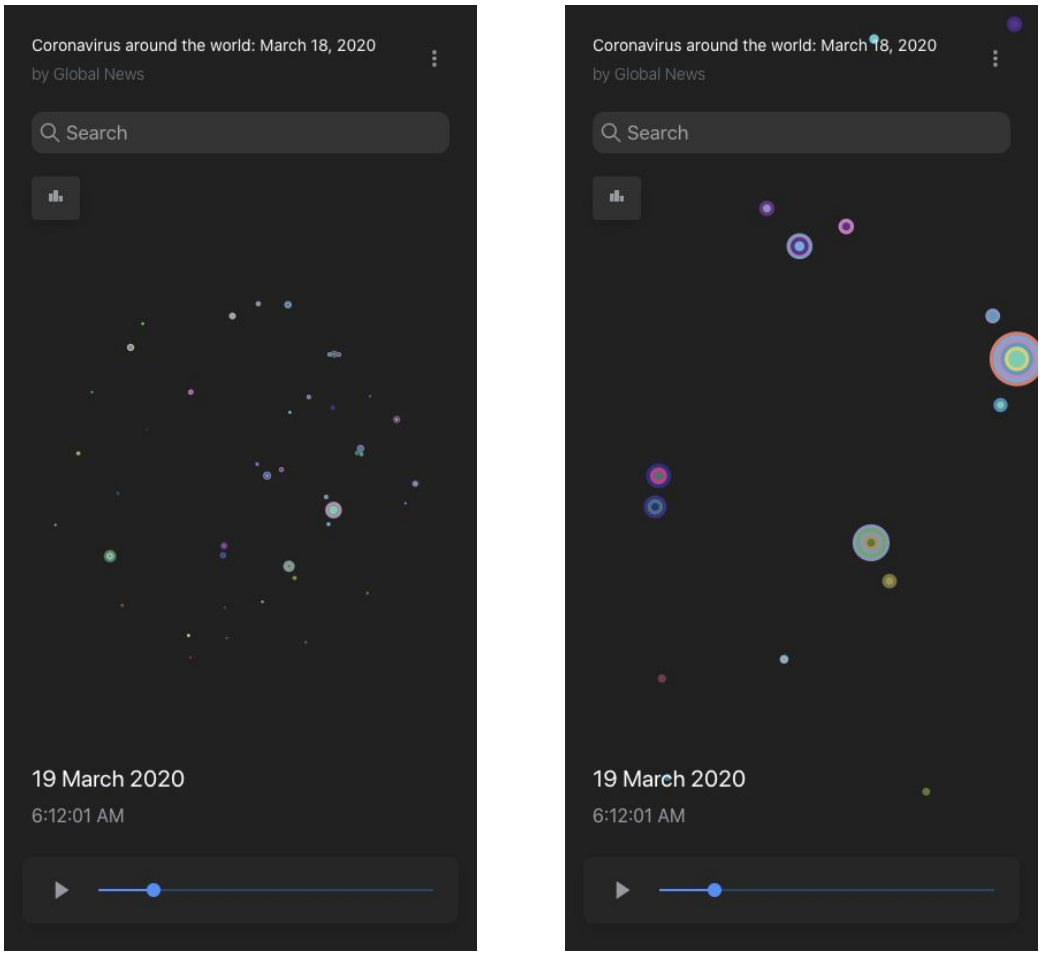


Figure 42(b): Viewing Canvas Page - Circles Representing Top-level Comment

Gradually, concentric circles are drawn on the top-level comment's center with a radius incremented by an offset, representing replies to that top-level comments (Figure 42(c)).

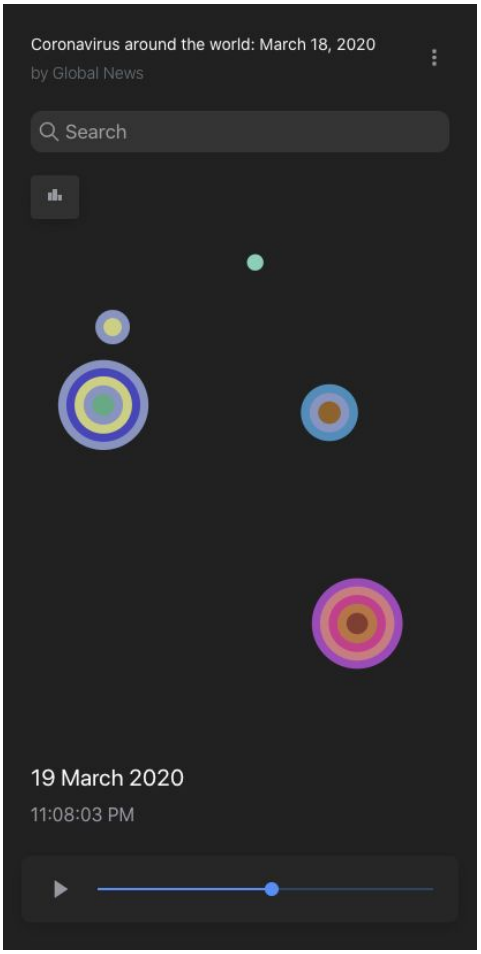


Figure 42(c): Viewing Canvas Page - Concentric Circles Representing Replies to Top-level Comment

Peripheral concentric circles are also shown with close proximity to some concentric circles, representing subset discussion threads within the parent's nucleic discussion thread (Figure 42(d)).

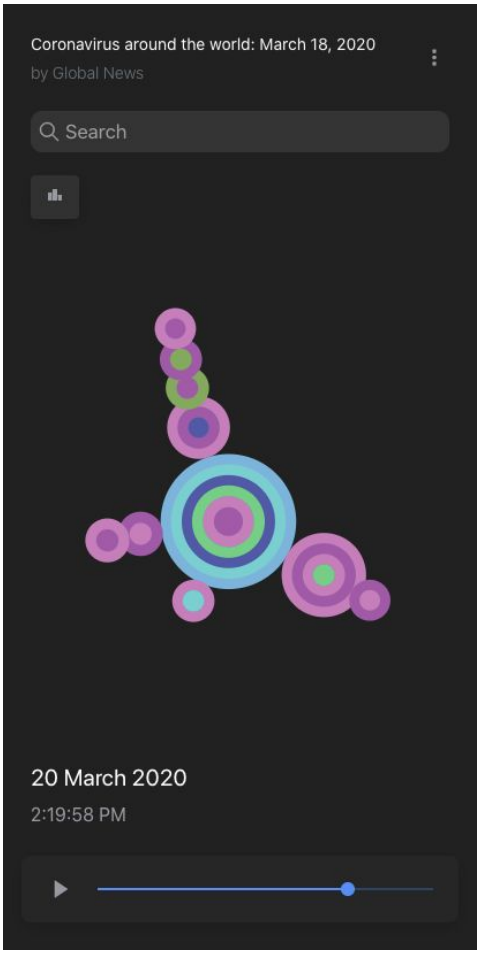


Figure 42(d): Viewing Canvas Page - Peripheral Concentric Circles Representing Subset Discussion Thread

7.3.3 Viewing Detailed Information

Clicking on a concentric circle displays its relevant information, including the contents, the author names, the number of likes and the respective dates the comments were published (Figure 43). There are also colored bars drawn on each comment item representing the comment like count normalized by a scale derived from the discussion thread it belongs to.

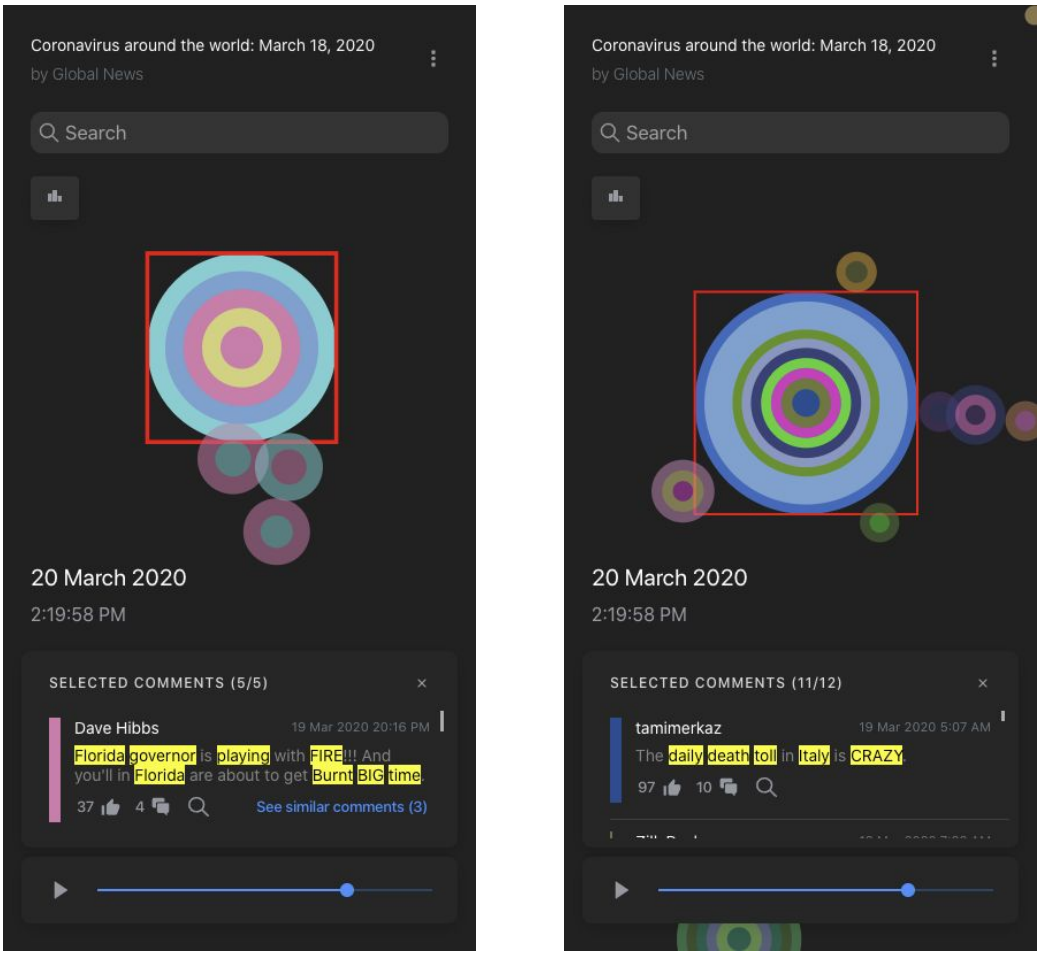


Figure 43: Viewing Detailed Information

To navigate to a certain comment with replies to it, clicking ‘View replies’ highlights and pans the canvas to the related concentric circle’s location (Figure 44(a)).

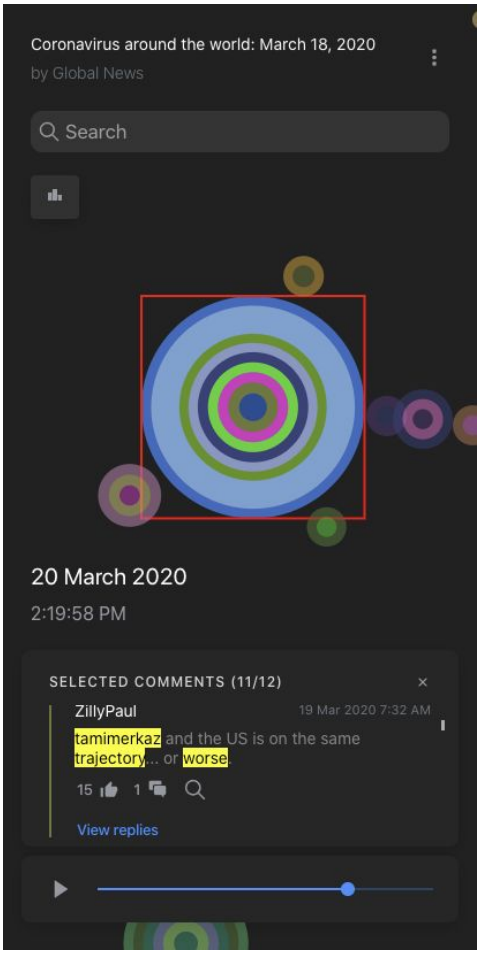


Figure 44(a): Utilizing Shortcuts to Navigate to Comment with Replies

Clicking ‘View as reply to ...’ pans the canvas back to the concentric circle’s location of the parent comment (Figure 44(b)).

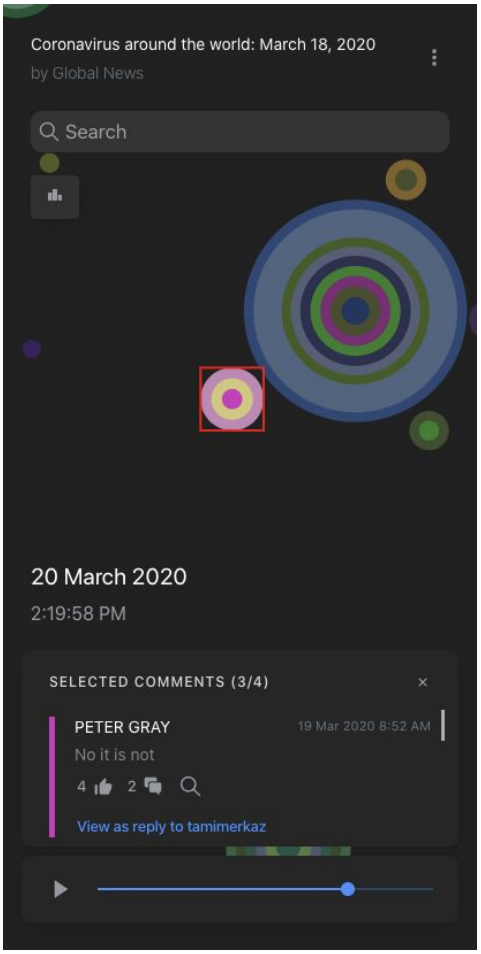


Figure 44(b): Utilizing Shortcuts to Navigate to Parent Comment

Keeping the concentric circle selected while the application is in play mode will dynamically add comments to the list and resize the selected concentric circle automatically.

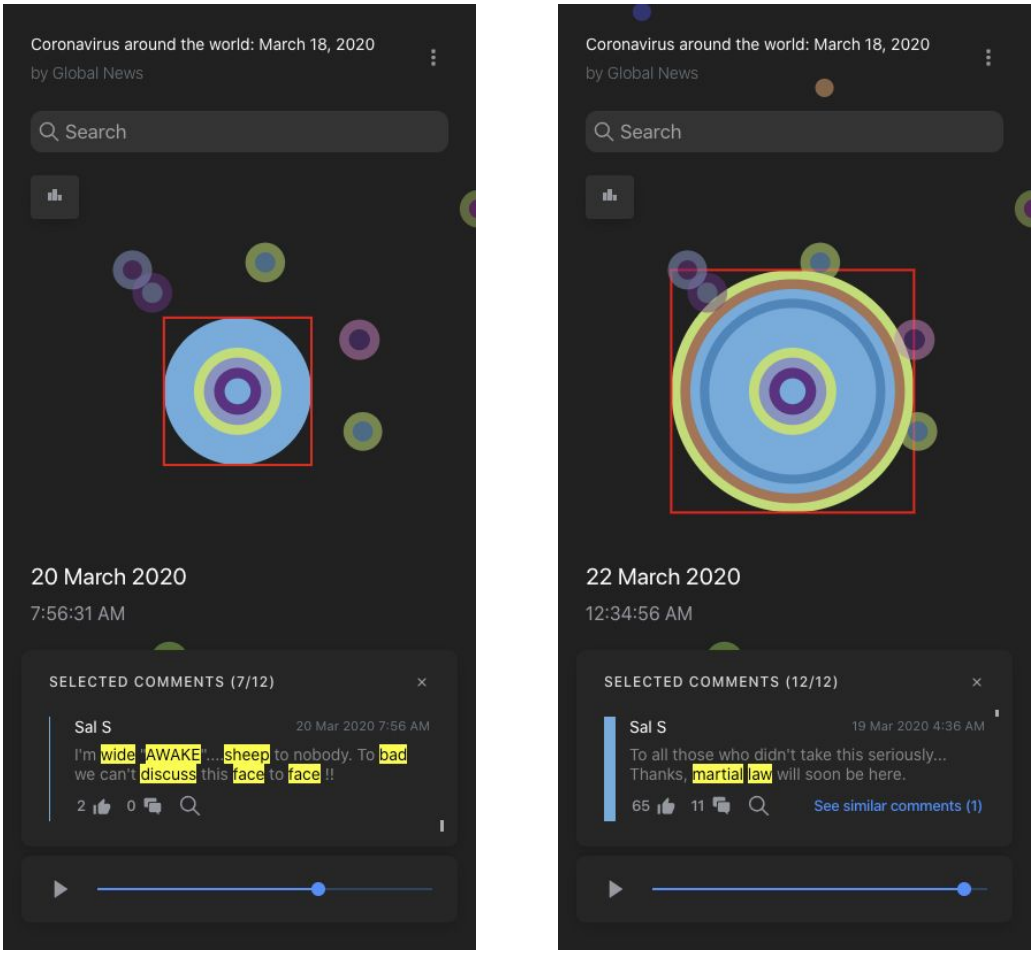


Figure 45: Selected Concentric Circle in Play Mode

7.3.4 Analyzing Evolution of Social Comments Posting Pattern

Upon clicking on the bar-chart button activated, the spectrum mode is activated. All concentric circles are immediately displayed and the timeline player controls are replaced with a bar-chart like visualization representing the number of comments published over time (Figure 46). This provides insights about changes to the posts's level of discussion activity from the date the first comment was published to the date of the latest comment extracted. It can be observed that the amount of comments decreases generally over time.

Adjusting the range of the time frame of the spectrum filters and highlights comments published within the time frame. In contrast, comments published within the said time frame are blurred out from the canvas.

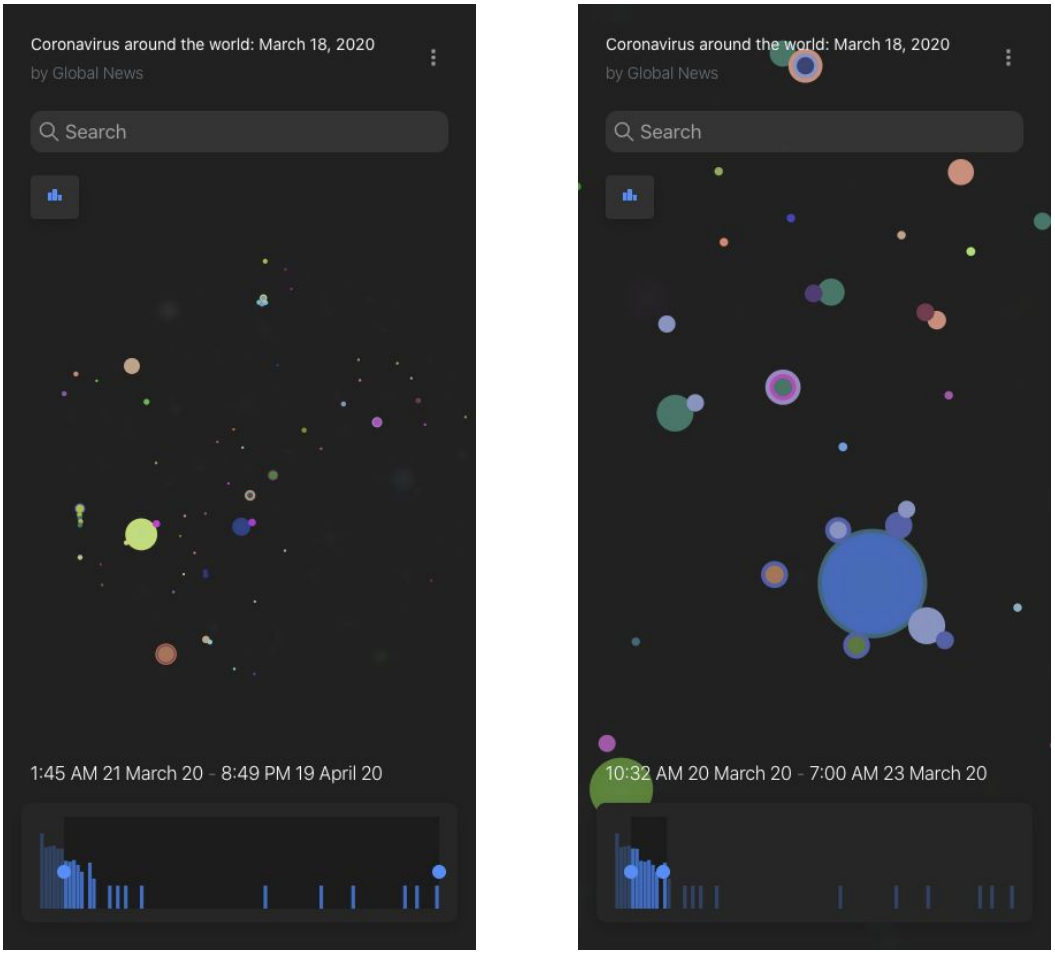


Figure 46: Analyzing Evolution of Social Comments Posting Pattern

7.3.5 Finding Similar Comments

Identified topics (e.g. “Florida”, “governor”, and “tourism”) within the textual content are highlighted in yellow (Figure 47). The same topics are internally used in matching other comments with similar topic distribution.

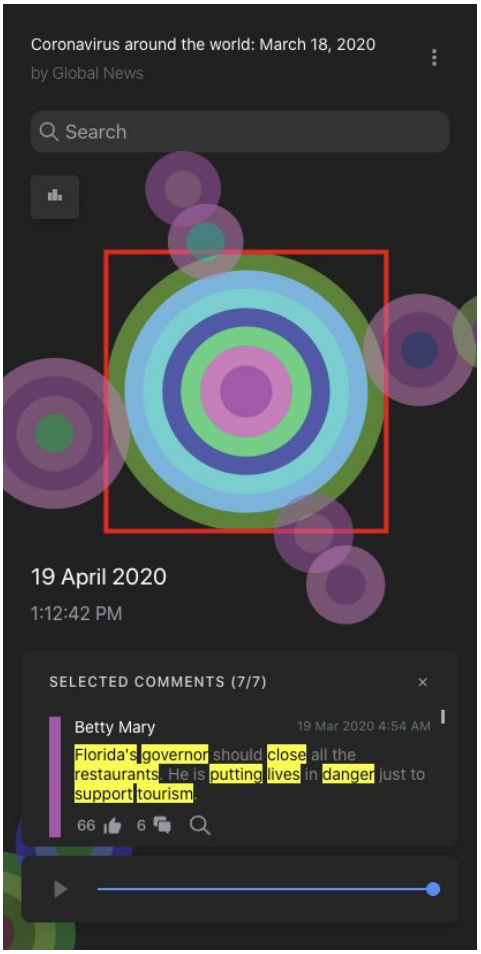


Figure 47: Identified Topics

Clicking on “See similar comments” will affix the selected comment at the top of the interface while the list of similar comments are listed below (Figure 48). On the canvas, comments that are not similar to the reference comment are made less visible, and matching similar comments are focused by rendering a white boundary box around it.

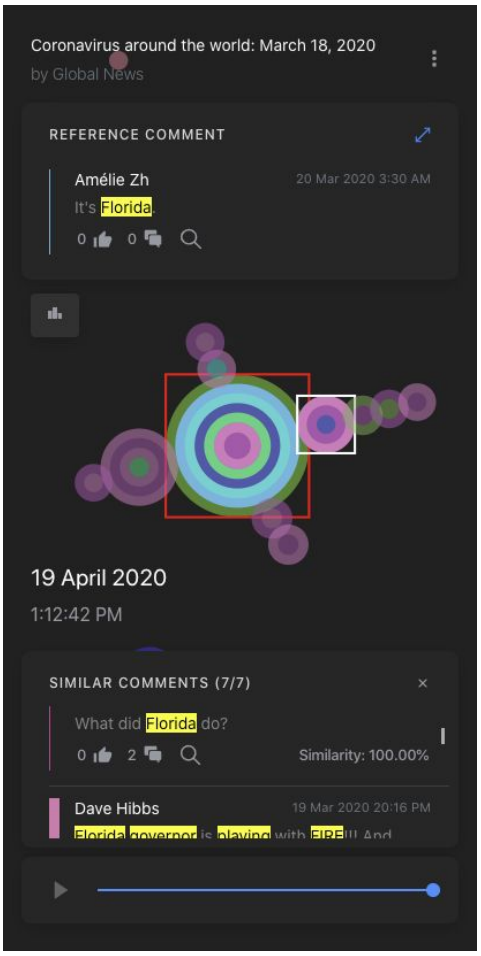


Figure 48: Finding Similar Comments

To aid user navigation, clicking on the zoom button (spyglass icon) on a similar comment on the list pans the canvas to its circle's position (Figure 49).

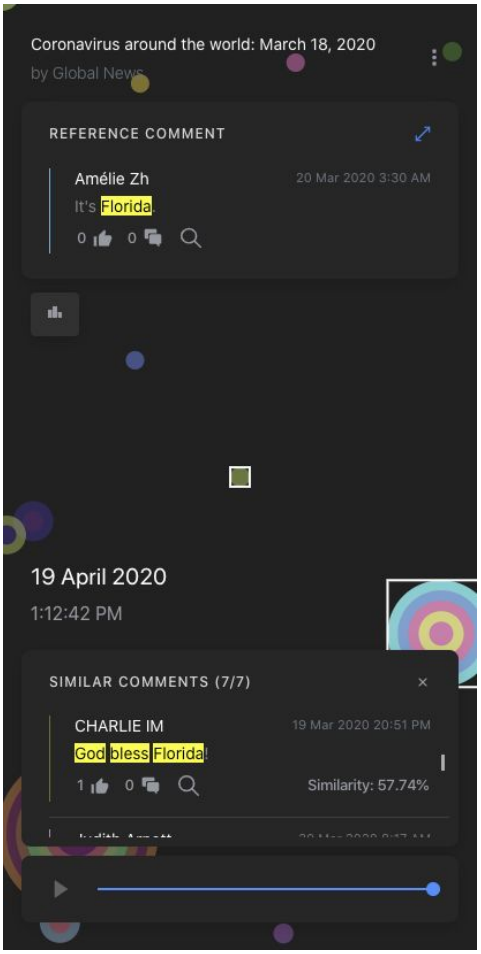


Figure 49: Checking Similar Comment

7.3.6 Searching for Keywords

When the keyword “Italy” is entered in the search bar, concentric circles that contain comments matching “Italy” are highlighted and surrounded with a white boundary box (Figure 50). Selecting one of the highlighted concentric circles displays its comments and highlights the matching keywords in blue.

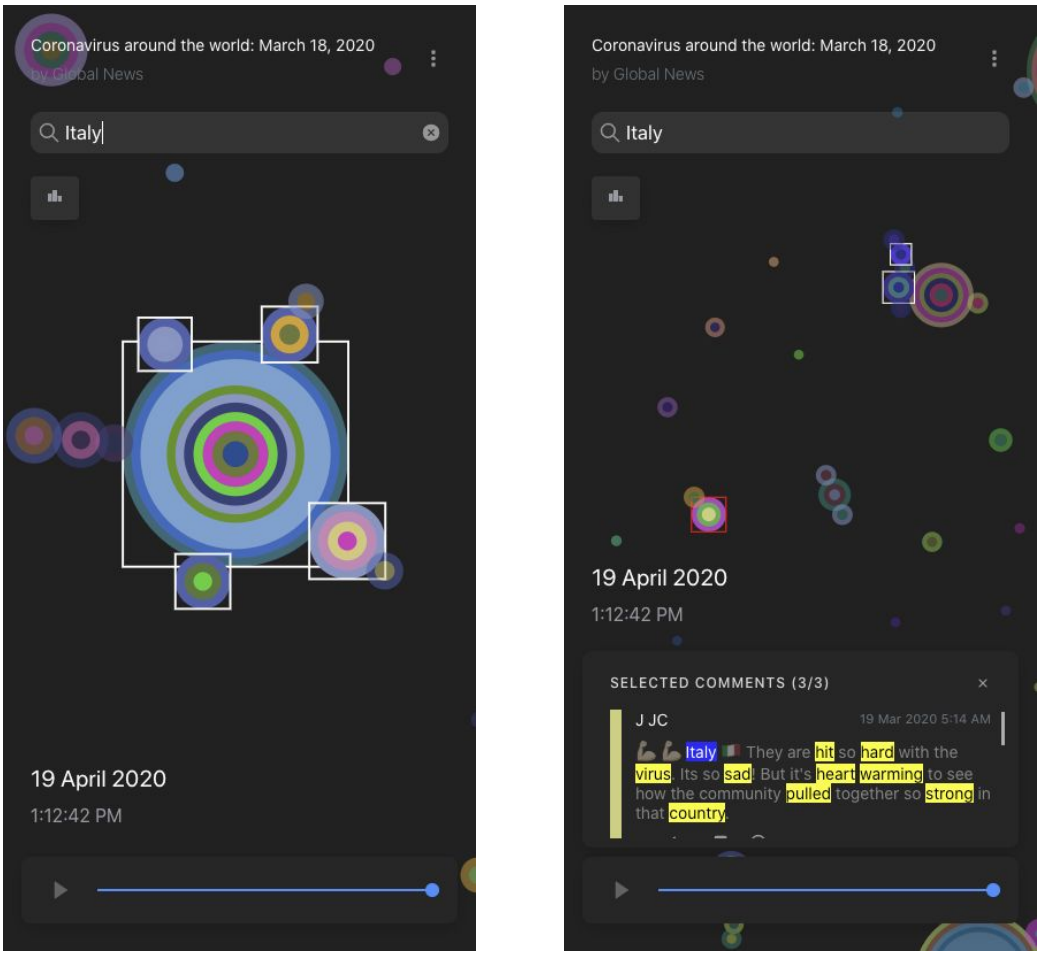


Figure 50: Searching for Keywords

8. Performance Study

The performance of an application is essential as it is directly linked to the user experience. In this chapter, the performance of KANDINSKY Mobile is evaluated with results from experiments and user study.

8.1 Performance of Different Modules

The performance evaluation of KANDINSKY Mobile focuses on two aspects - speed and functionality. To ensure the diversity of video posts, 50 YouTube videos under 5 different topics are randomly selected for testing (Table 15). All selected videos have varying numbers of comments within the range of hundreds.

Topic	**Number of YouTube Videos**
Equality	10
Environment	10
Technology	10
Arts	10
Education	10

Table 15: Test Datasets

All features provided by the application are tested for the processing time and response correctness. Each feature for each video post is tested for 10 trials to mitigate outlier effects and obtain the average performance. Each trial is conducted without any other applications running on the machine and without any other browsing activities. Cache is also refreshed for each run. The average processing time of the 10 trials is then used as the speed indicator. All experiments are run under the same environment as summarized in Table 16.

Machine	Pixel 2 XL
Operating System / Version	Android OS / Version 10.0
Processor	Qualcomm MSM8998 Snapdragon 835
Memory	4 GB
Internet Connection / Speed	Wifi Connection / 92.2 Mbps

Table 16: Test Environment

In performing the experiment, timestamped logs are printed in the console when stages start and end. This allows us to see the cost breakdown for both stages when running data preparation and potentially identify bottlenecks in the process. At the end of this subsection, the performance of the overall data preparation step is compared with the scenario when the data prepared is already stored in cache.

8.1.1 Extraction

For this section, extraction refers to the stages involved in making the data ready for visualisation. This primarily includes data retrieval which includes fetching of post data, paginated fetching of comments and replies data, and mapping of these entities into internal schema models as described in section 4.2 Model Mapping. The performance of model mapping is measured together with data retrieval due to the piped nature of both processes.

From the results displayed below, we can observe a positive correlation between the number of comments and the time it takes to complete the data retrieval processes. This can be explained by the paginated nature of fetching post comments from Youtube's API. The more the comments a post has, the higher the number of commentary page requests we need to make to retrieve the complete list of comments.

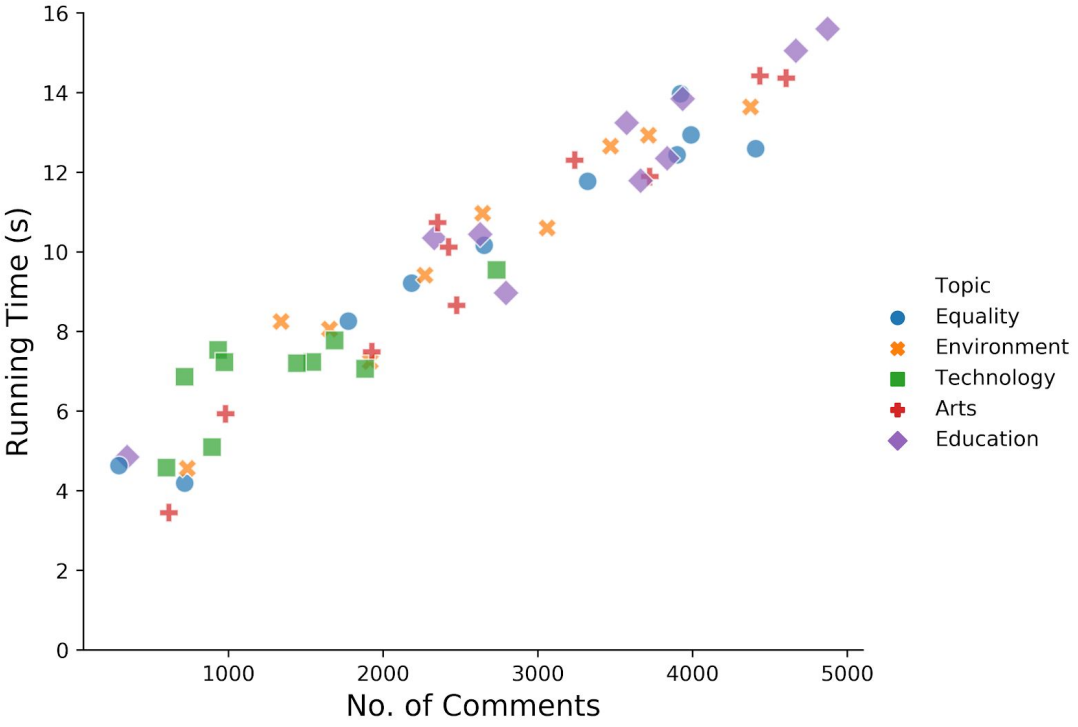


Figure 51: Average Processing Time for Extraction

8.1.2 Representation

The analytical preprocessing stage includes topic modelling and matching comments together with other comments that have similar topic distributions. The complexity of fixed-iteration LDA per iteration is theoretically O(N, T) where N is the number of words in all the comments and T represents the number of topics which is fixed at 1 in the implementation. Thus, the time complexity should be linearly proportional to the number of words in the comments. However, in matching comments with others that have similar topic distributions, this process scales at an exponential rate relative to the number of comments since every comment topic distribution needs to be compared with all other comment topic distributions.

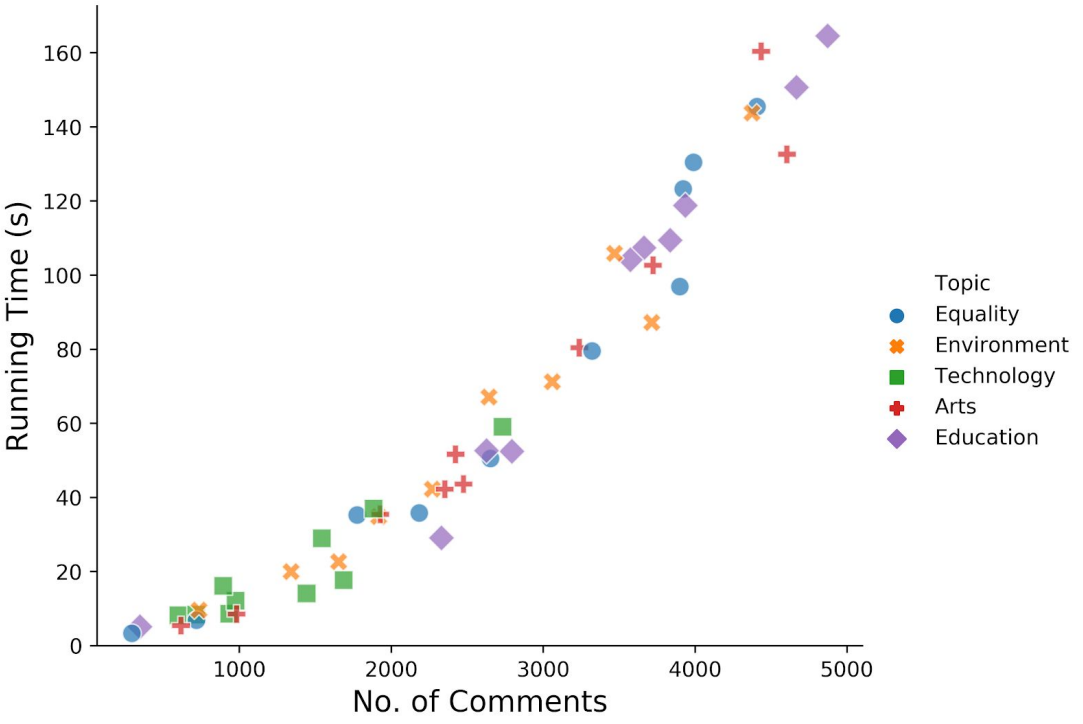


Figure 52: Average Processing Time for Representation

8.1.3 Data Layout

The canvas page is generated by performing a preparatory step to transform the comments into a canvas-specific schema, and by drawing the actual elements on the canvas through DOM element insertion and running the custom force-directed simulation to position the inserted elements. The time complexity is theoretically O(C) where C represents the total number of comments. This matches the observed trends in the experiments as shown in Figure 53 and 54, where the running time is linearly proportional to the comment count. The response of all tests render correct visualization as expected.

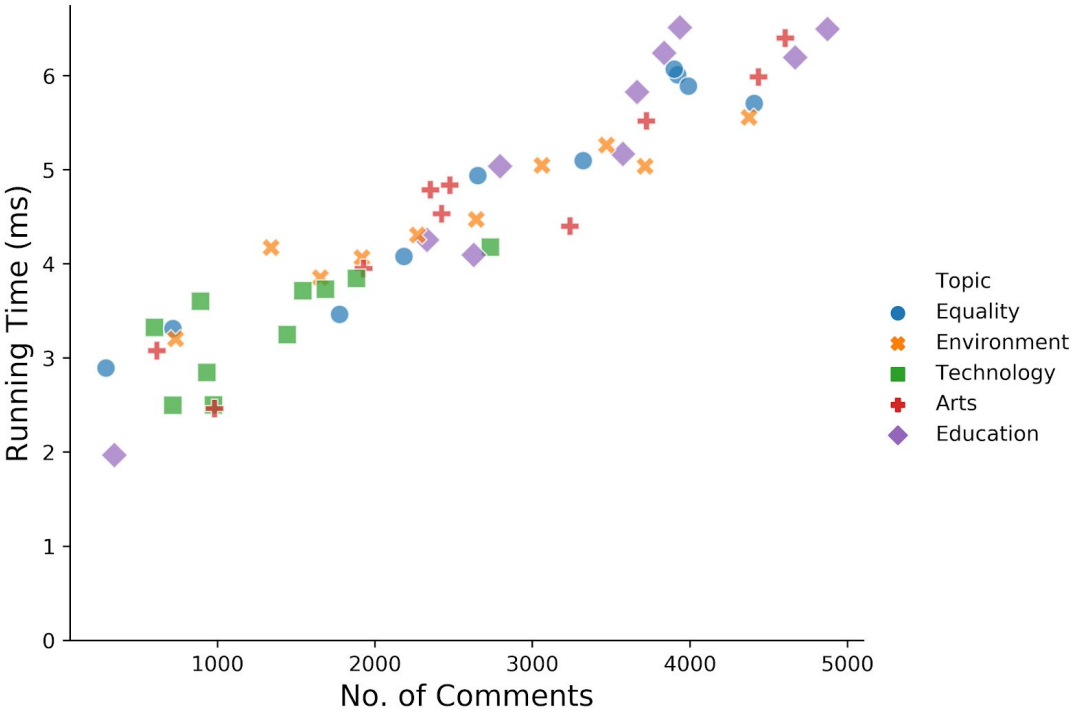


Figure 53: Average Processing Time for Canvas Preparation

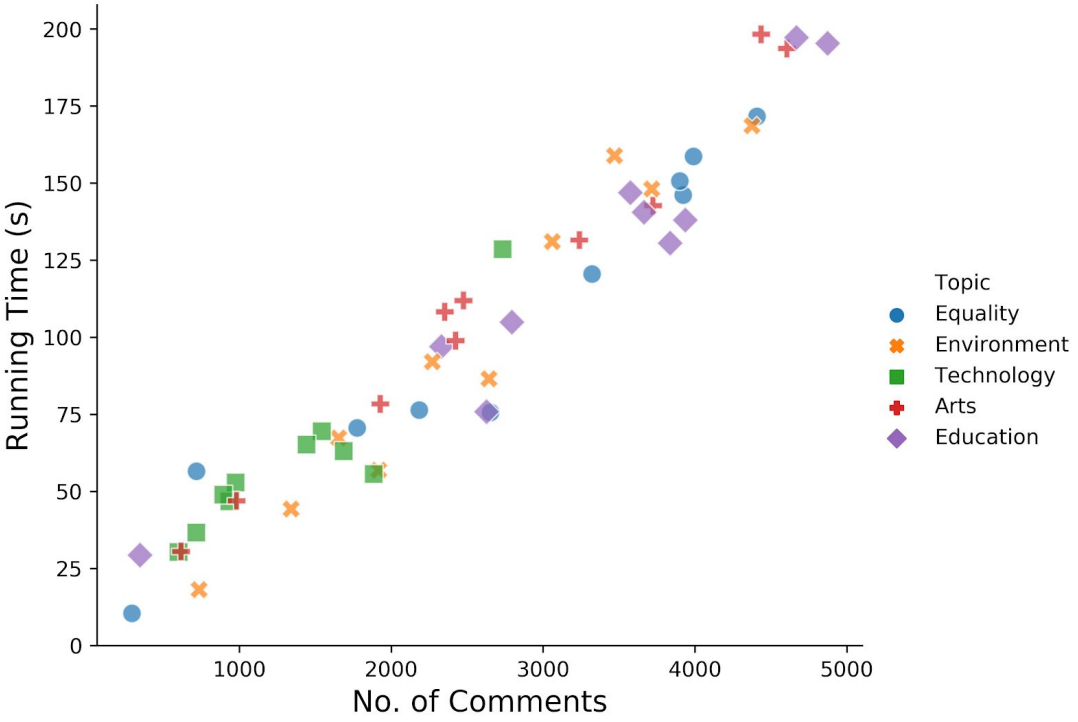


Figure 54: Average Processing Time for Canvas Drawing

For this part of the evaluation, we compare the performance of the custom force-directed simulation layout as implemented in the application vs a generic force-directed simulation layout without the custom forces as discussed in section 5.2 Algorithm. The generic force-directed simulation layout is only influenced by minimal forces to achieve the force-directed simulation layout described in section 5.3.2 Force-Directed Simulation Layout. From the results below, we can see that the custom force-directed simulation layout performs almost twice as bad as the generic force-directed simulation layout. This can be explained by the heavy calculations required to apply the additional forces on the simulation itself.

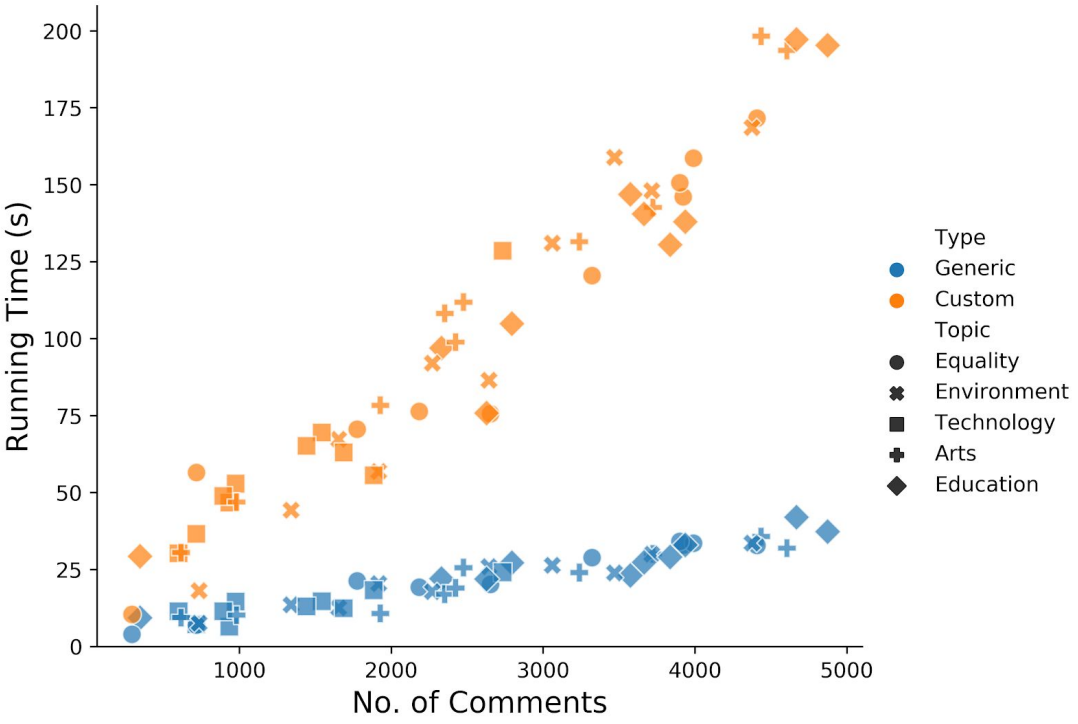


Figure 55: Average Processing Time for Canvas Generation with Custom (Circles Layout) vs Generic Force-Directed Simulation Layout

8.1.3 Caching

The overall time it takes to perform extraction, representation and data layout is compared with a typical revisit scenario when the data preparation has already been done and the processed data is already saved in the device's storage. To help portray the difference between the two cases, the running time values are adjusted in the logarithmic scale.

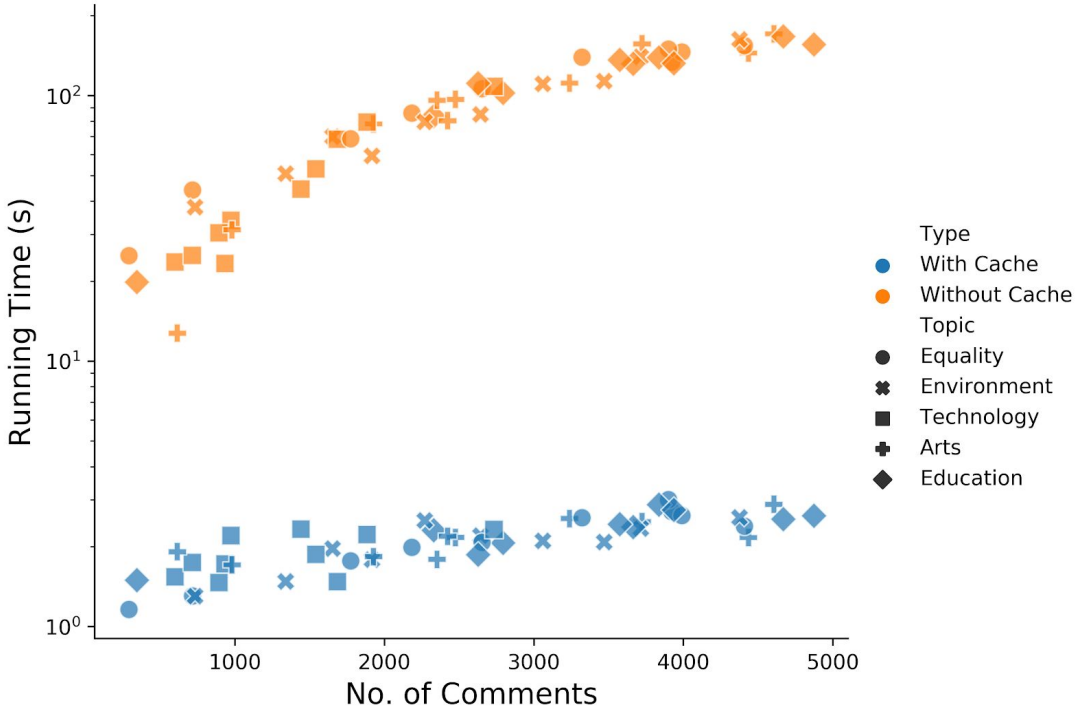


Figure 56: Average Processing Time with Caching vs Without Caching

8.1.4 Keywords Search

The search function searches for comments that contain a given set of keywords and highlight concentric circles that contain comments matching the given keywords. For all experiments, a set of keywords with varying lengths related to the respective topics is searched. The time complexity for the regular expression algorithm applied for keyword search is theoretically O(L) where L represents the total length of textual content of all the comments. This matches the observed trends in the

experiments as shown in Figure 57, where the running time is roughly linearly proportional to the comment count which is roughly linearly proportional to the total length of comments.

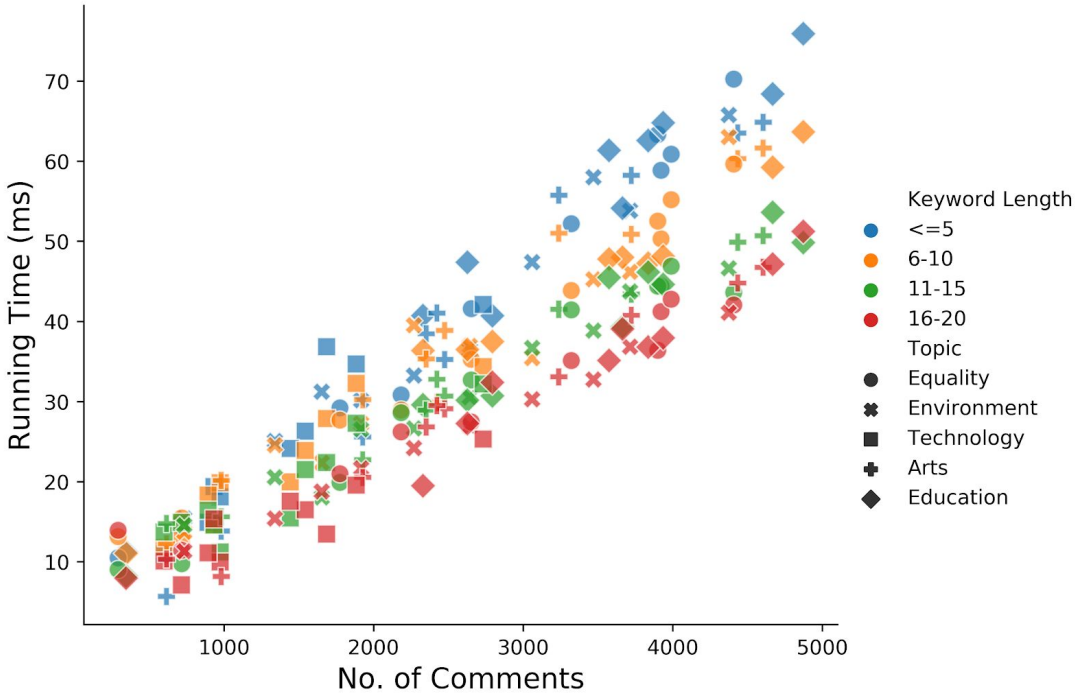


Figure 57: Average Processing Time for Keywords Search

8.2 User Study

To evaluate the usability of KANDINSKY, a usability study was conducted to evaluate whether KANDINSKY provides an effective, efficient, useful and enjoyable experience for users. A total of 20 people participated in the user experience (UX) study. All participants used their own mobile phone to watch the several YouTube videos, and then used KANDINSKY Mobile on their own mobile phone to perform data visualization and analysis on the comments associated with that video. Finally, they were asked to answer a survey which questions can be found in Appendix 1. The survey focused on feedback on the data layout, UI design and task efficiency, as compared to the current mobile view. This chapter assesses KANDINSKY Mobile

based on the survey results. The detailed survey responses can be found in Appendix 2.

As shown in Figure 58 and 59, the participants of the survey are from equally distributed gender and age groups. This demographic diversity ensures the coverage and reliability of the usability study.

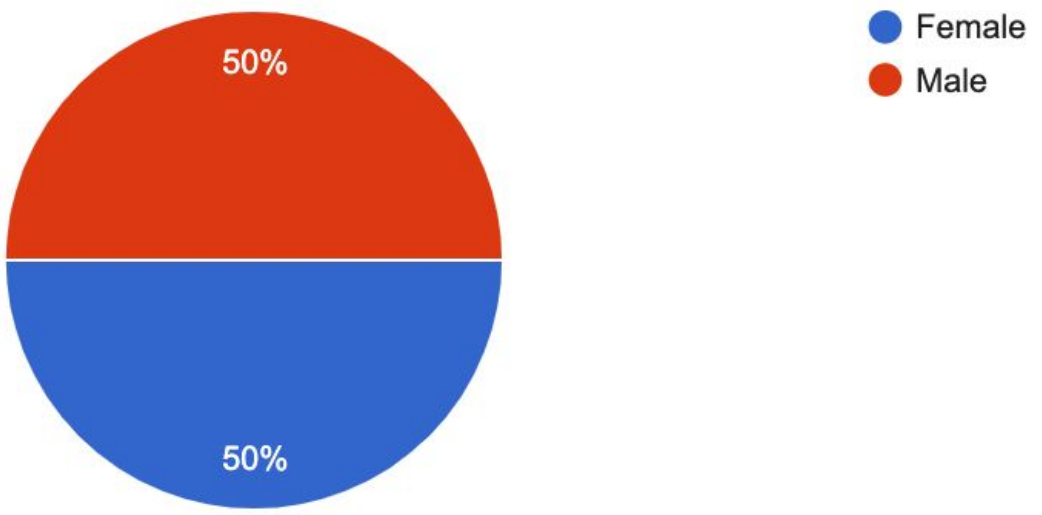


Figure 58: Gender Distribution

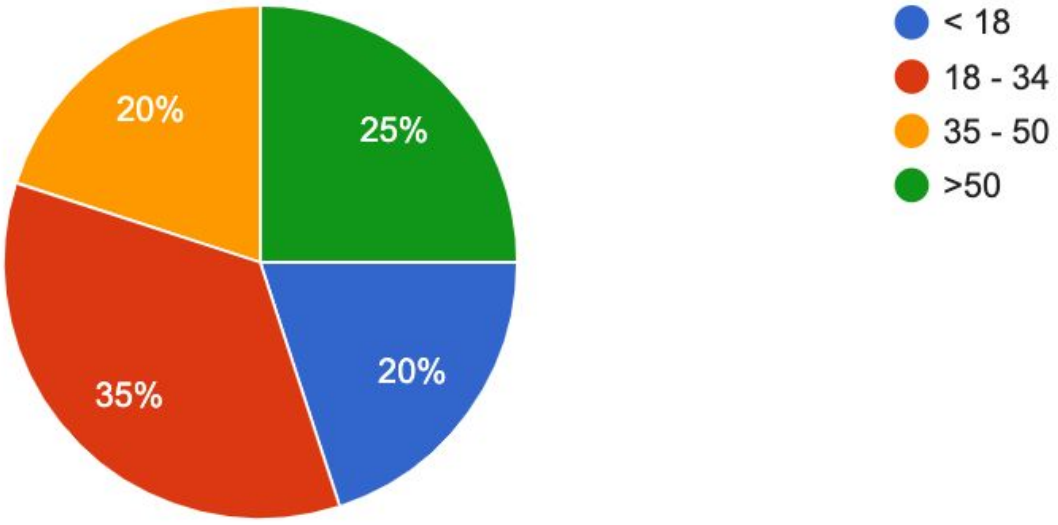


Figure 59: Age Group Distribution

8.2.1 Data Layout

Participants agree that KANDINSKY Mobile provides a better data layout than the current mobile view to a large extent, with an average score of 9.3 out of 10. This high score shows the design of the data layout is well accepted by users and preferred by users as compared to the current mobile view.

8.2.2 UI Design

Participants agree that KANDINSKY Mobile presents a better UI design than the current mobile view to a large extent, with an average score of 9.3 out of 10. This high score shows the UI design is well accepted by users and preferred by users as compared to the current mobile view.

8.2.3 Task Efficiency

Participants agree that KANDINSKY Mobile helps them understand and analyse the comments more efficiently than the current mobile view to a large extent, with an average score of 9.45 out of 10. This high score shows the system achieves the purpose of providing a more efficient tool for people to understand and analyse social discussions.

8.2.4 Ease of Learning

The ease of learning of the system achieves an average score of 9.15 out of 10. This high score could be attributed to the minimalism design as mentioned by several participants and the incorporation of human computer interaction design in the UI.

8.2.5 Usefulness

When being asked whether they read comments of social media posts, correspondents give an average score of 8.55 out of 10, showing that people are interested in the information from social discussions.

The system scores 8.35 out of 10 in terms of overall usefulness. Breaking down into individual features, all five features (Canvas View, Details-in-demand, Spectrum Filter, Keyword Search and Similar Comments) scores above 8 out of 10 for usefulness, with the feature of Canvas View receiving the highest score. This shows the features provided by KANDINSKY Mobile are well accepted by the users in terms of its usefulness in helping them understanding and analysing social comments.

8.2.6 Performance

The integration of the system receives a high average score of 9.05 out of 10, while the inconsistency and lagginess have a rating of 1.7 and 1.6 out of 10 respectively. This shows the system is perceived to be well-integrated and efficient by the user.

8.2.7 Overall Satisfaction

The participants rate the system at 9.25 out of 10 in terms of overall satisfaction, 7.8 out of 10 in terms of willingness to use frequently, and 8 out of 10 in terms of intention to recommend. Users generally have a satisfied experience when using KANDINSKY Mobile.

The minimalism design and innovative visualization design are the favourite characteristics of KANDINSKY Mobile by most users.

8.2.8 Area of Improvements

Participants also provide constructive recommendations and ideas for future improvement. Users want the application to extend support for more platforms and languages, to allow exportation of information such as the canvas screenshot, and to provide comment summary and sentiment analysis. These are all taken into considerations on the discussion of future work in the next chapter.

9. Conclusions and Future Work

The ubiquitous usage and high user engagement of social media generates voluminous information from social conversations, providing a rich source of information on public opinions. However, there is still no such a tool to facilitate effective and in-depth analysis and understanding of the discussions. Moreover, the improving capability and growing popularity of mobile devices make it necessary to develop such a tool for mobile phone users to deliver greater convenience. This project has developed KANDINSKY Mobile, a mobile-based end-to-end visualization application, for supporting the user on analysing copious and complicated social conversations. Based on usability evaluation, the visualization design and human computer interaction design of the application are both intuitive and satisfying for users.

There are several directions of future work for KANDINSKY Mobile. Sentiment analysis of social comments can be incorporated to rate the general public feedback on a certain anchor post and to provide another angle for more in-depth analysis. Besides, the application can be extended to support analysis of comments in more languages and for more social media platforms, such as Instagram.

References

Ahlberg, C., Williamson, C., & Shneiderman, B. (1992, June). Dynamic queries for information exploration: An implementation and evaluation. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 619-626).

Aleman, A. C., Wang, M., & Schaeffel, F. (2018). Reading and myopia: contrast polarity matters. Scientific reports, 8(1), 1-8.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, 3(Jan), 993-1022.

Boring, S., Ledo, D., Chen, X. anthony, Marquardt, N., Tang, A., & Greenberg, S. (2012). The fat thumb. Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services Companion - MobileHCI 12. doi: 10.1145/2371664.2371711.

Chapman, C. (2018, March 27). Exploring the Gestalt Principles of Design. Retrieved from https://www.toptal.com/designers/ui/gestalt-principles-of-design.

DataReportal. Social Media Users By Platform. https://datareportal.com/social-media-users.

Donath, J., Karahalios, K., & Viegas, F. (1999). Visualizing conversation. Journal of computer-mediated communication, 4(4), JCMC442.

Easterby, R. S. (1970). The perception of symbols for machine displays. Ergonomics, 13(1), 149-158.

Engdahl, B., Köksal, M., & Marsden, G. (2005, April). Using treemaps to visualize threaded discussion forums on PDAs. In CHI'05 extended abstracts on Human factors in computing systems (pp. 1355-1358).

Fernando, N., Loke, S. W., & Rahayu, W. (2013). Mobile cloud computing: A survey. Future generation computer systems, 29(1), 84-106.

Few, S., & Edge, P. (2017). Data visualization effectiveness profile. Perceptual Edge, 1-11.

Giller, V., Melcher, R., Schrammel, J., Sefelin, R., & Tscheligi, M. (2003, September). Usability evaluations for multi-device application development three example studies. In International Conference on Mobile Human-Computer Interaction (pp. 302-316). Springer, Berlin, Heidelberg.

Hoober, S., & Berkman, E. (2012). Designing mobile interfaces: Steven Hoober, Eric Berkman. Beijing: OReilly.

Hoober, S. (2017, March 6). Design for Fingers, Touch, and People, Part 1. Retrieved from https://www.uxmatters.com/mt/archives/2017/03/design-for-fingers-touch-and-people-part-1.php.

Hoober, S. (2017, July 10). Design for Fingers, Touch and People, Part 3. Retrieved from https://www.uxmatters.com/mt/archives/2017/07/design-for-fingers-touch-and-people-part-3.php.

Huang, K. Y. (2009, October). Challenges in human-computer interaction design for mobile devices. In Proceedings of the World Congress on Engineering and Computer Science (Vol. 1, No. 6, pp. 1-6).

ITE Publications, Measuring digital development - Facts and figures (2019). https://www.itu.int/en/ITU-D/Statistics/Documents/facts/FactsFigures2019.pdf.

Johnson, J. (2020). Designing With The Mind In Mind: simple guide to understanding user interface design guidelines. S.l.: MORGAN KAUFMANN PUBLISHER.

Jones, M., Buchanan, G., & Thimbleby, H. (2002, September). Sorting out searching on small screen devices. In Proceedings of International Conference on Mobile Human-Computer Interaction (pp. 81-94). Springer, Berlin, Heidelberg.

Jones, M., Buchanan, G., Thimbleby, H., & Marsden, G. (2000, May). User interfaces for mobile web devices www9 mobile workshop position paper. InProceedings of 9th International World Wide Web Conference.

Kamba, T., Elson, S. A., Harpold, T., Stamper, T., & Sukaviriya, P. (1996, April). Using small screen space more efficiently. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 383-390).

Kandinsky, W. Squares with concentric circles. http://www.kandinskypaintings.org/squares-with-concentric-circles/.

Kandinsky, W. Several circles. https://en.wikipedia.org/wiki/File:Vassily_Kandinsky,_1926_-_Several_Circles,_Gugg_0910_25.jpg.

Lidwell, W., Holden, K., & Butler, J. (2010). Universal principles of design: 125 ways to enhance usability, influence perception, increase appeal, make better design decisions, and teach through design. Beverly, MA: Rockport Publishers.

Lui, C., Bhowmick, S. S., & Jatowt, A. (2019, July). KANDINSKY: Abstract Art-Inspired Visualization of Social Discussions. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 1345-1348).

Marcus, A., Cowan, W. B., & Smith, W. (1989, March). Color in user interface design: functionally and aesthetics. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 25-27).

Mathur, A., Schlotfeldt, B., & Chetty, M. (2015, September). A mixed-methods study of mobile users' data usage practices in South Africa. In Proceedings of the

2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (pp. 1209-1220).

Mobile Usability Testing: Fat Finger Syndrome. (2019, July 3). Retrieved from https://ux247.com/mobile-usability-testing-fat-finger-syndrome/

Noirhomme-Fraiture, M., Randolet, F., Chittaro, L., & Custinne, G. (2005, May). Data visualizations on small and very small screens. In Proceedings of the 11th International Symposium on Applied Stochastic Models and Data Analysis.

Parhi, P., Karlson, A. K., & Bederson, B. B. (2006). Target size study for one-handed thumb use on small touchscreen devices. Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI 06. doi: 10.1145/1152215.1152260

Punchoojit, L., & Hongwarittorrn, N. (2017). Usability studies on mobile user interface design patterns: a systematic literature review. Advances in Human-Computer Interaction, 2017.

Raneburger, D., Alonso-Ríos, D., Popp, R., Kaindl, H., & Falb, J. (2013, September). A user study with GUIs tailored for smartphones. In IFIP Conference on Human-Computer Interaction (pp. 505-512). Springer, Berlin, Heidelberg.

Shneiderman, B. (1996, September). The eyes have it: A task by data type taxonomy for information visualizations. In Proceedings 1996 IEEE symposium on visual languages (pp. 336-343). IEEE.

Smith, J. (2019), Smart Phone Use Linked to Eye Strain and Decreased Tear Production. https://www.naturalhealthresearch.org/smart-phone-use-linked-to-eye-Strain-and-decreased-tear-production.

Stone, D. L. (2005). User interface design and evaluation. Amsterdam: Morgan Kaufmann.

Todorovic, D. (n.d.). Gestalt principles. Retrieved from http://www.scholarpedia.org/article/Gestalt_principles?__hstc=77520074.36a0ddae8e24bce7.

UX for Mobile: The Rise of Fat-Finger Design. (n.d.). Retrieved from https://www.outsystems.com/blog/posts/ux-for-mobile-fat-finger-design/.

UX Movement (2020, February 19). Why You Should Never Use Pure Black for Text or Backgrounds. https://uxmovement.com/content/why-you-should-never-use-pure-black-for-text-or-backgrounds.

Wang, L., & Sajeev, A. S. M. (2007, January). Roller interface for mobile device applications. In Proceedings of the eight Australasian conference on User interface-Volume 64 (pp. 7-13). Australian Computer Society, Inc..

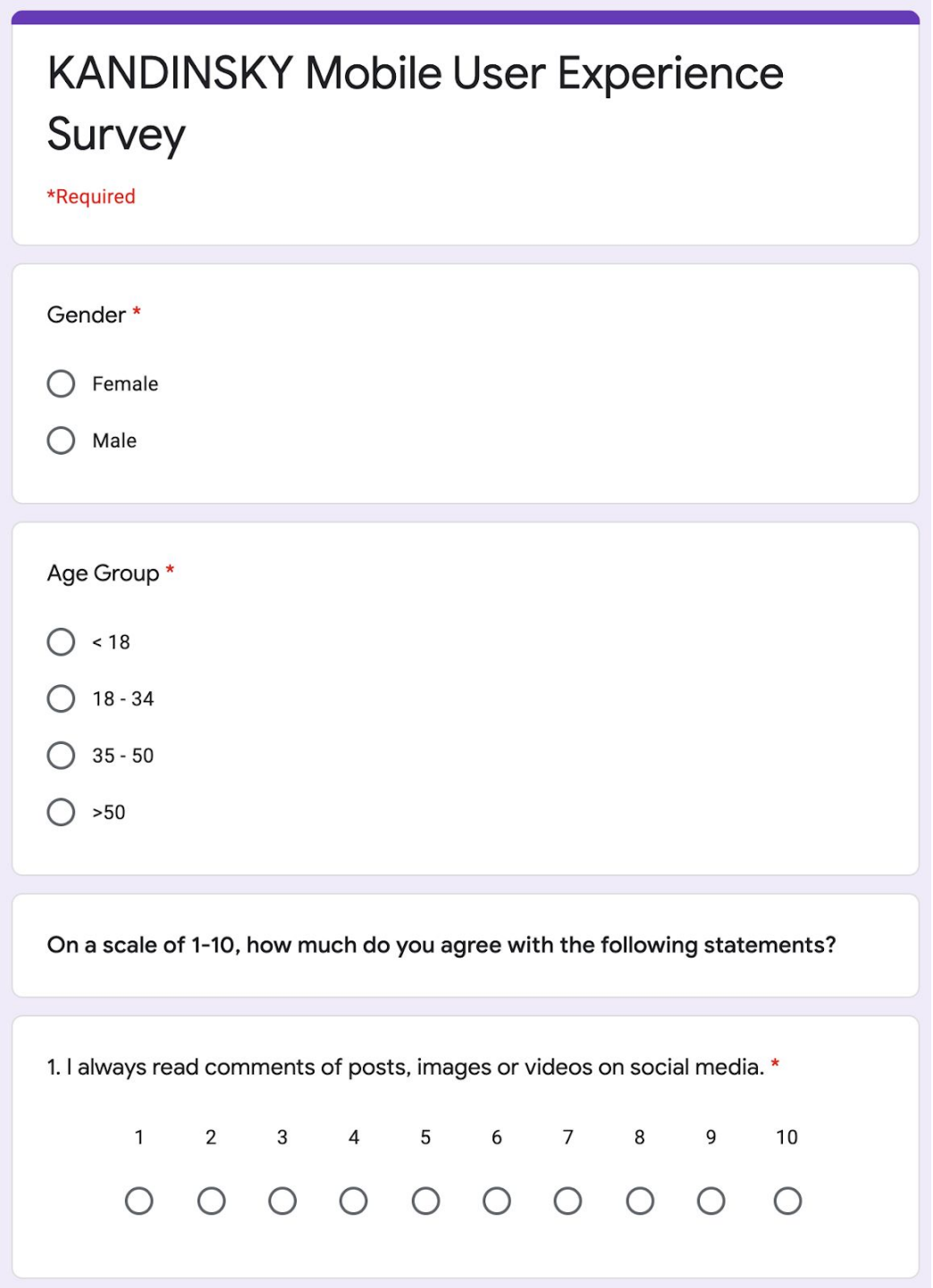
Wertheimer, M. (1938). Laws of organization in perceptual forms. In W. D. Ellis (Ed.), A source book of Gestalt psychology (p. 71–88). Kegan Paul, Trench, Trubner & Company.

Wurman, Richard Saul, Information Anxiety, Doubleday, New York (1989).

Xiong, R., & Donath, J. (1999, November). PeopleGarden: creating data portraits for users. In Proceedings of the 12th annual ACM symposium on User interface software and technology (pp. 37-44).

Appendices

Appendix 1: KANDINSKY Mobile User Experience Survey



KANDINSKY Mobile User Experience Survey

*Required

Gender *

- Female
- Male

Age Group *

- < 18
- 18 - 34
- 35 - 50
- >50

On a scale of 1-10, how much do you agree with the following statements?

1. I always read comments of posts, images or videos on social media. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

2. I enjoy using the system. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

3. I find the system easy to use. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

4. The system is useful for me. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

5. I would like to use the system frequently. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

6. I find the functions in the system well integrated. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

7. I think there is inconsistency in the system. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

8. I find the system laggy. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

9. I will recommend the system to my friends. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

10. I find the data layout better than the current mobile view. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

11. I find the user interface design better than the current mobile view. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

12. The system helps me understand and analyse the comments more efficiently than the current mobile view. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

13. The feature of Canvas Visualization (bird's-eye view) is useful. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

14. The feature of Details-on-demand (display detailed information of a comment) is useful. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

15. The feature of Spectrum Filter is useful. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

16. The feature of Keyword Search is useful. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

17. The feature of Similar Comments is useful. *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

Open-Ended Questions

1. What do you like most about KANDINSKY? *

Your answer

2. What do you like least about KANDINSKY? *

Your answer

3. What other features do you think the system should provide? *

Your answer

4. Do you have any recommendations on the future improvement? *

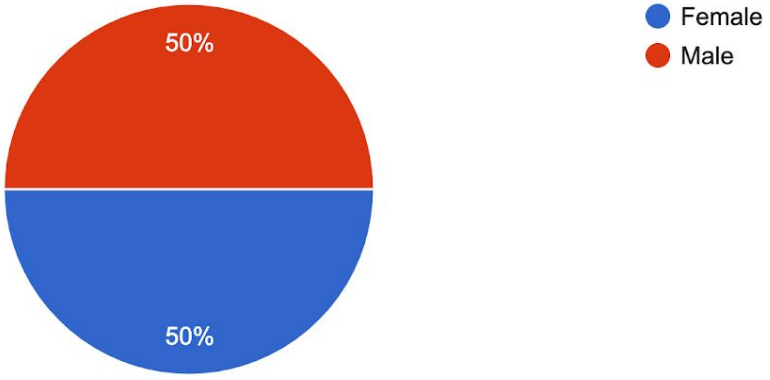
Your answer

Submit

Appendix 2: User Experience Survey Results

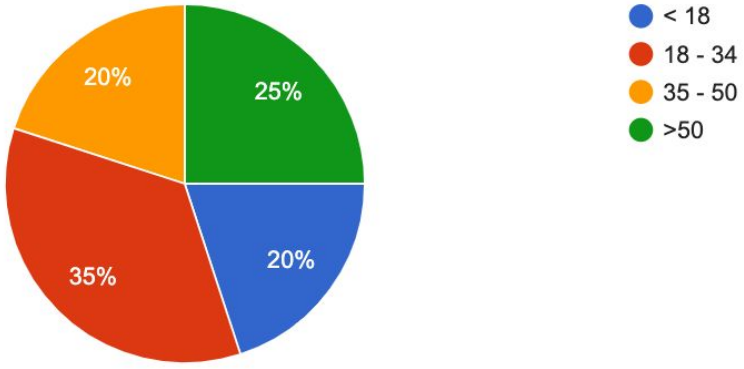
Gender

20 responses



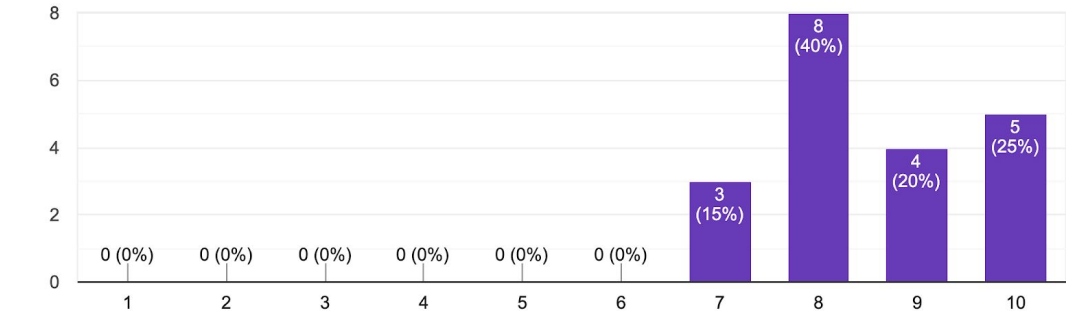
Age Group

20 responses



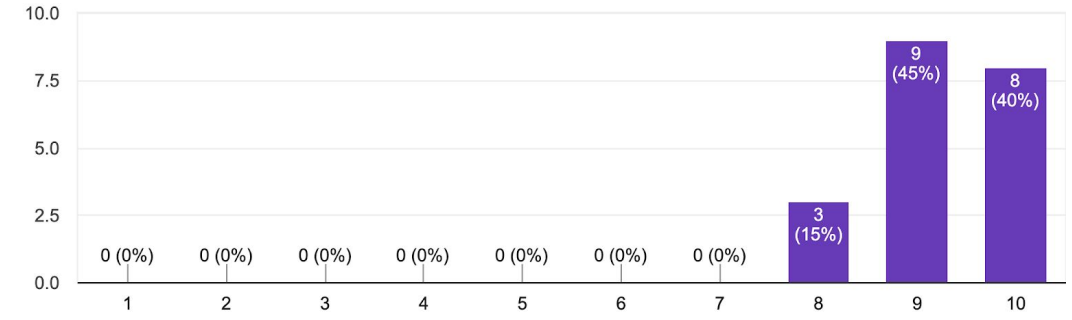
1. I always read comments of posts, images or videos on social media.

20 responses



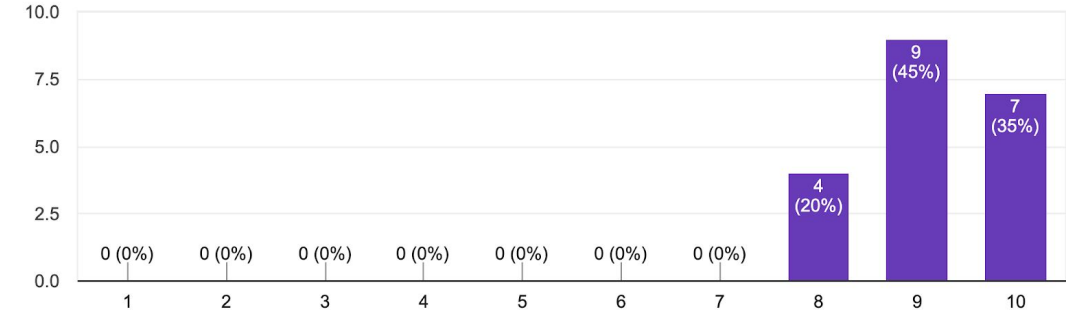
2. I enjoy using the system.

20 responses



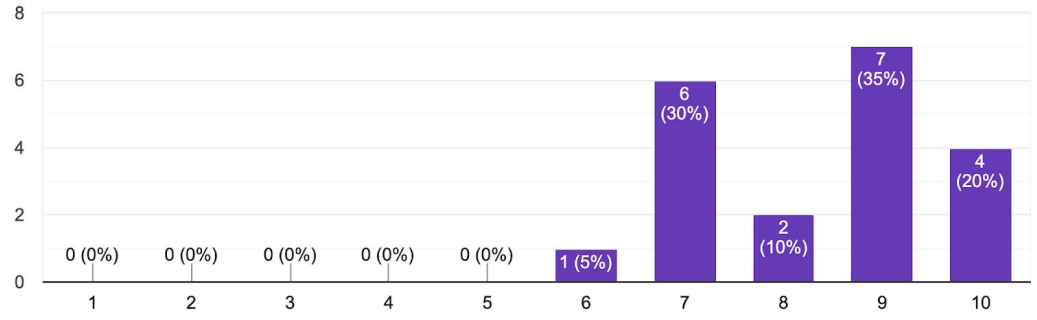
3. I find the system easy to use.

20 responses



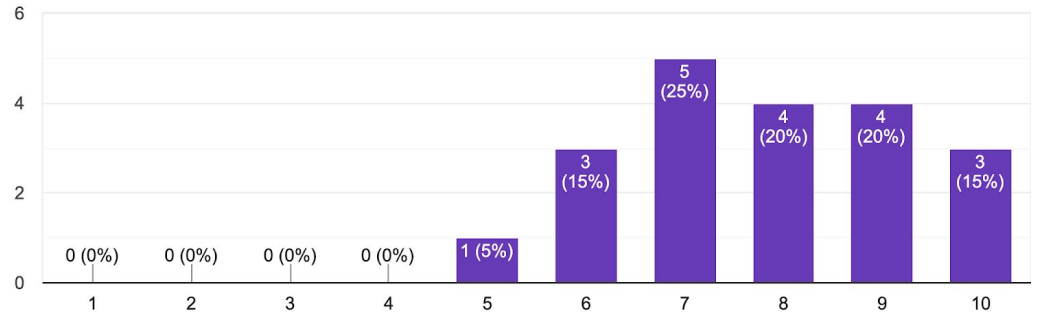
4. The system is useful for me.

20 responses



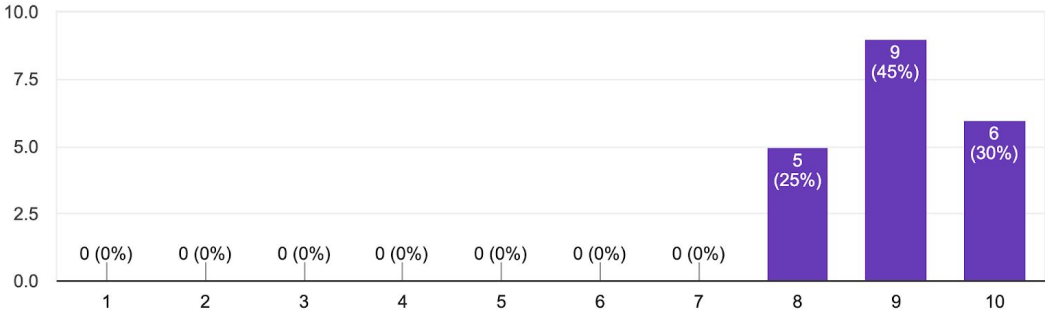
5. I would like to use the system frequently.

20 responses



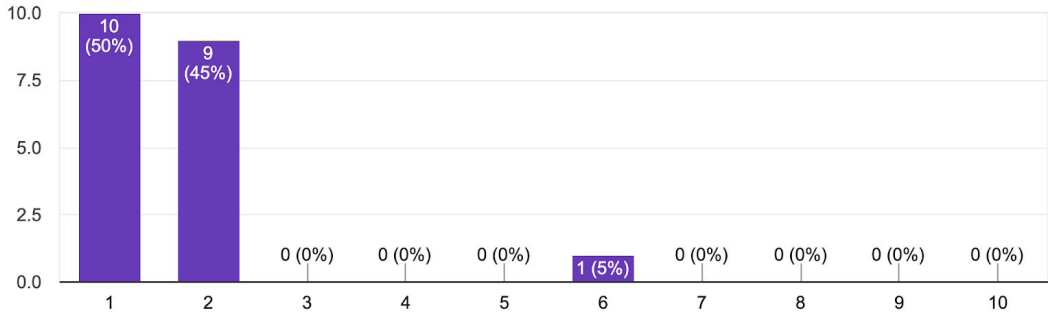
6. I find the functions in the system well integrated.

20 responses



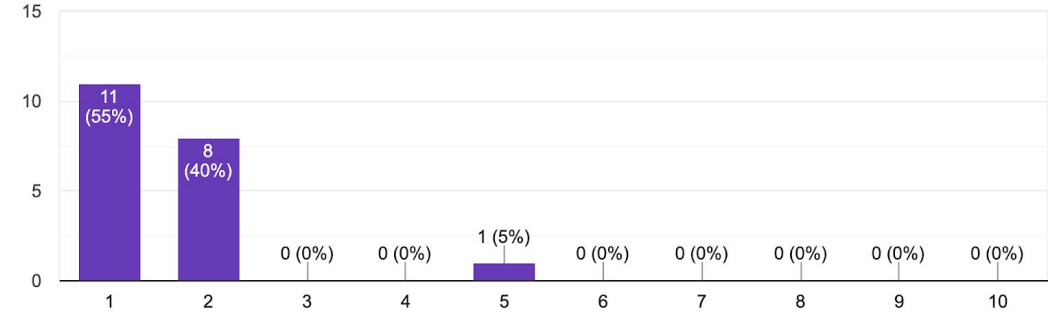
7. I think there is inconsistency in the system.

20 responses



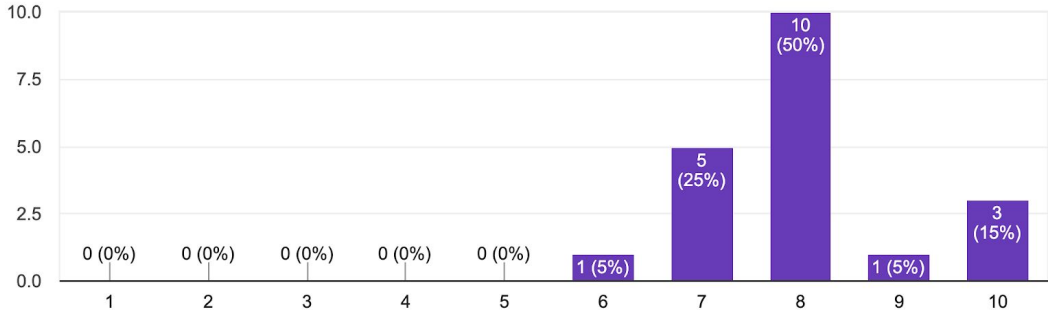
8. I find the system laggy.

20 responses



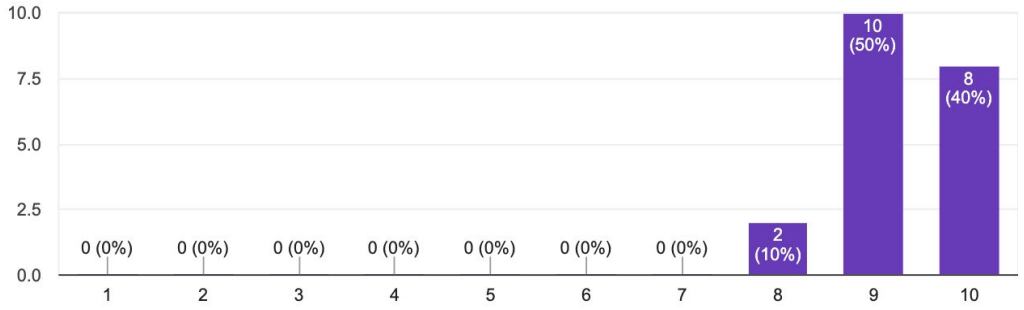
9. I will recommend the system to my friends.

20 responses



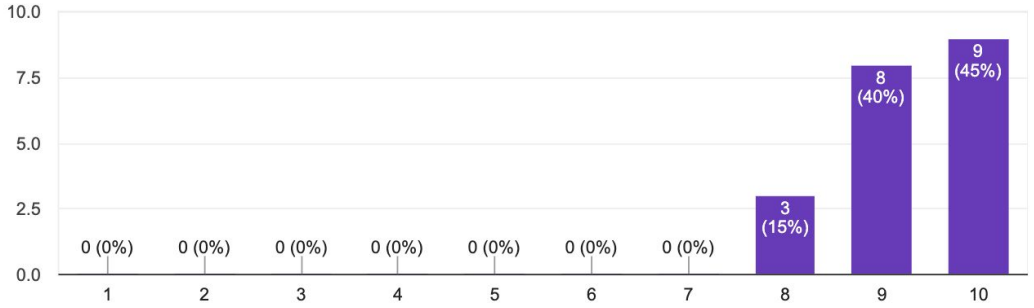
10. I find the data layout better than the current mobile view.

20 responses



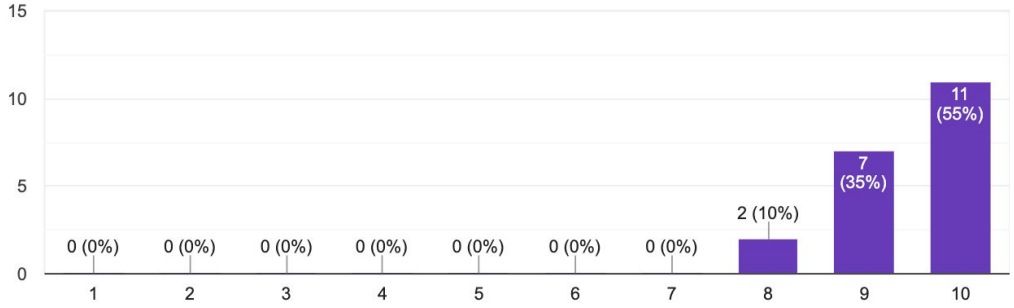
11. I find the user interface design better than the current mobile view.

20 responses



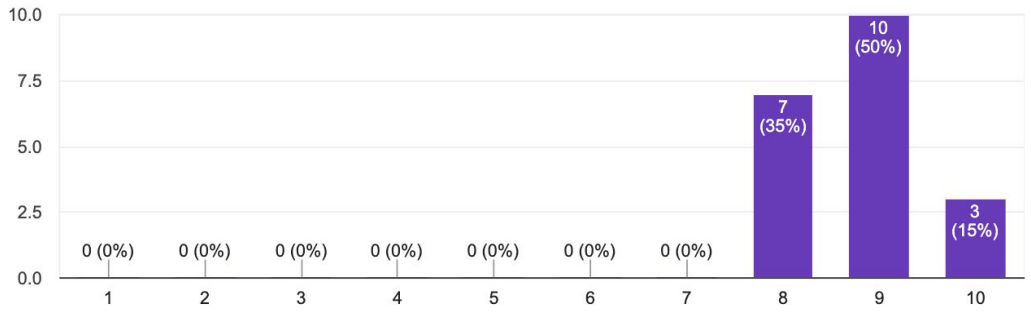
12. The system helps me understand and analyse the comments more efficiently than the current mobile view.

20 responses



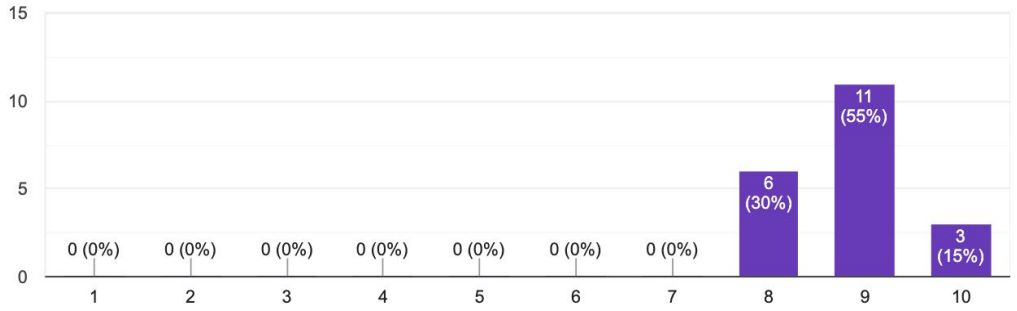
13. The feature of Canvas Visualization (bird's-eye view) is useful.

20 responses



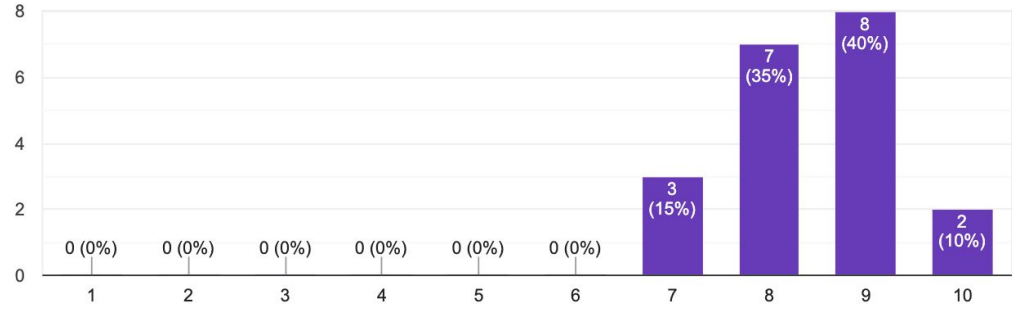
14. The feature of Details-on-demand (display detailed information of a comment) is useful.

20 responses



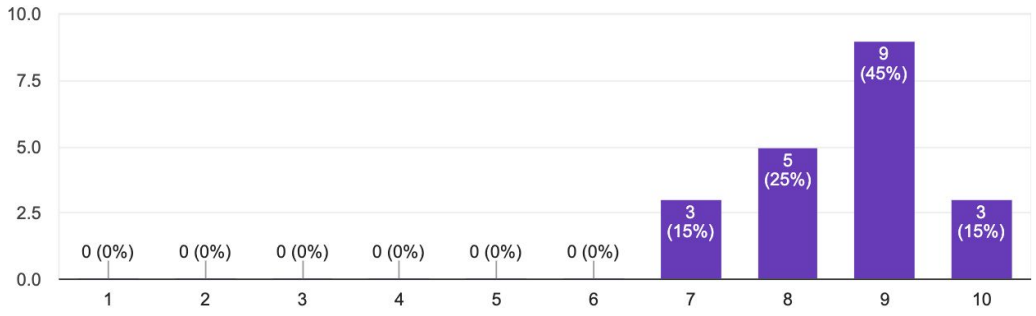
15. The feature of Spectrum Filter is useful.

20 responses



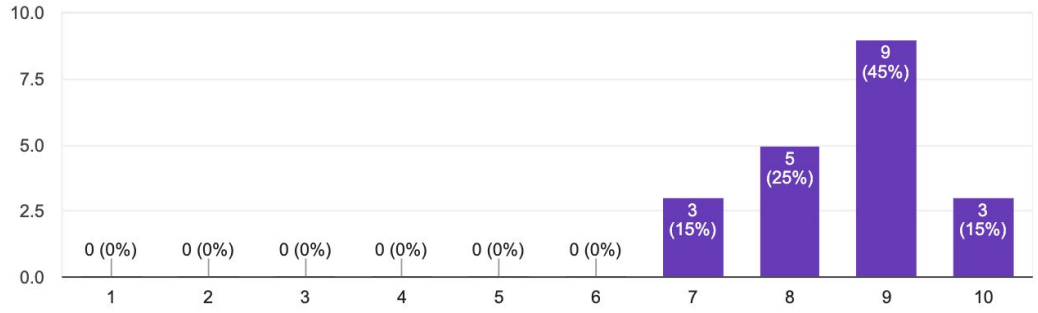
16. The feature of Keyword Search is useful.

20 responses



17. The feature of Similar Comments is useful.

20 responses



1. What do you like most about KANDINSKY?

20 responses

Easy to use
Minimalism design
the features and functions are very useful
The UI looks nice
Design
Nice design
Keyword Search function
Innovative visualisation of displaying comments
Nice interface
Many ways to analyze comments
Innovative design
Cool UI
na
UI
Cool design
Artistic visualization
Good interface design

2. What do you like least about KANDINSKY?

20 responses

na
Nil
NA
nil
Cannot export information
Cannot export screenshot
Only support YouTube

3. What other features do you think the system should provide?

20 responses

na
Nil
Exportation of information
Support Instagram
Allow export of image for the canvas
Sentiment analysis
Support Facebook and Instagram
Comments summary
Support other languages such as Chinese
Support more platforms such as fb, twitter and insta
More platforms please
Support for other Facebook and Instagram
Comment summary
nil
Export data
Support Facebook

4. Do you have any recommendations on the future improvement?

20 responses

na
Nil
NA
nil
Add exportation of information
Add to Google Play Store
Allow export of image for the canvas
More platforms and languages
Support more languages