T-Cal: Understanding Team Conversation Data with Calendar-based Visualization

Siwei Fu[1] **Jian Zhao**[2] **Hao-Fei Cheng**[3] **Haiyi Zhu**[3] **Jennifer Marlow**[2]
[1]Hong Kong University of Science and Technology [2]FXPAL [3]University of Minnesota
sfuaa@cse.ust.hk {zhao, marlow}@fxpal.com {cheng635, zhux0449}@umn.edu

ABSTRACT
Understanding team communication and collaboration patterns is critical for improving work efficiency in organizations. This paper presents an interactive visualization system, T-Cal, that supports the analysis of conversation data from modern team messaging platforms (e.g., Slack). T-Cal employs a user-familiar visual interface, a calendar, to enable seamless multi-scale browsing of data from different perspectives. T-Cal also incorporates a number of analytical techniques for disentangling interleaving conversations, extracting keywords, and estimating sentiment. The design of T-Cal is based on an iterative user-centered design process including interview studies, requirements gathering, initial prototypes demonstration, and evaluation with domain users. The resulting two case studies indicate the effectiveness and usefulness of T-Cal in real-world applications, including daily conversations within an industry research lab and student group chats in a MOOC.

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces

Author Keywords
Team messaging platforms; conversation data; calendar-based visualization; visual analytics.

INTRODUCTION
Team messaging platforms (e.g., Slack [3] and Microsoft Teams [1]) are becoming increasingly popular in organizations as a means of team communication. These platforms are designed with similar features including public and private channels, direct messaging, instant chats, etc. Analyzing data generated by such platforms (e.g., conversation logs) is critical to understanding team communicative and coordinative practices. It can provide insights to learn how the successful team operation is constituted [6], and allow for gleaning relational links among team members which are significant to the effectiveness of information exchange [49].

However, the analysis of team conversation data is nontrivial due to two main reasons. First, the growing volume of such data and its heterogeneous nature (e.g., containing textual, temporal, and team member information) hinder users from grasping meaningful and critical patterns due to information overload. Second, multiple topical points may appear simultaneously during team communication, and individual conversational threads often interleave, making it challenging for users to follow and digest the content.

Visualization has been proven effective in exploring conversations in an intuitive and interactive manner, especially in the context of *threaded* conversation data where coherent posts are bundled up in one place (e.g., emails). Some work aims at presenting the structure of a thread to help users understand both structural and temporal patterns of threads [29, 23], and others employ analytical techniques to facilitate the investigation of topics and sentiment of threads [20, 26, 30]. Although powerful in their applications, these approaches are not adaptable to the team messaging platforms where conversational threads are interleaving. Visualization techniques targeted at *non-threaded* conversation data are under-exploited. As far as we know, some research has only been done in providing visual feedback of real-time chatting [9, 17, 31] and understanding transcribed spoken conversations [20, 21]. Little attention has been paid to sufficiently address the challenges in analyzing team collaboration or communication.

To fill the gap, in this paper we present a design study to explore interactive visualizations for investigating team conversation data. Our primary contribution is the design and implementation of T-Cal, a visual analytics system assisting users with the discovery of patterns in team communication and activities. T-Cal borrows a user-familiar interface, that is, a calendar, that allows users to explore massive message logs at different time scales, such as year, month, week, and day. We further employ advanced analytical methods to disentangle interleaving conversations into different threads, and summarize their content from different aspects. For example, we extract keywords from each thread, and infer sentiment for each message. We also contribute a novel visual design, ThreadPulse, inspired by electrocardiograms to visualize conversational threads. The design is able to illustrate both overall activity trends and detailed information of threads (e.g., keywords and authors of each message). Further, it is scalable and can display relatively long and intensive threads.

We situated our design study with conversation data from Slack channels [3], but we believe that the T-Cal design and our results are generalizable to other conversation datasets and team messaging platforms (e.g., Microsoft Teams [1])

CHI 2018, April 21–26, 2018, Montréal, QC, Canada.

http://dx.doi.org/10.1145/3173574.3174074

because their system functions, forms of data, and use-cases are very similar. We followed an iterative user-centered design process and involved users of different backgrounds such as team managers, human resource specialists, and knowledge workers. Two in-depth case studies demonstrate the effectiveness of T-Cal with real-world datasets and applications, including conversation logs within an industry research lab and communication of student groups in a MOOC.

RELATED WORK

Our work is related to visual analytics techniques for threaded and non-threaded conversation data, and more generally, temporal text corpora.

Visualization of Threaded Conversation data

The word *threaded* refers to the bundling up of a specific message and its replies in one place. Threaded conversation data stores reply relationships explicitly. Such data is commonly seen in communication platforms such as forum discussions, email messages, and blog posts.

Visualization of threaded conversations has received considerable attention over the past few years. Various approaches have been proposed to visualize the complex hierarchical replying structures, as seen in Conversation Thumbnail [50], TreeTable [35], tldr [34] and Content-centered Discussion Map [55]. Some other works [29, 44] aim to demonstrate both the hierarchical structure of threads and the temporal information of each entity. Recently, Thread River incorporates visual aggregation and focus+context techniques to extend the scalability of Thread Arcs [23]. However, the aforementioned approaches do not focus on depicting the general distribution of conversations and details of each message at the same time, which is supported by ThreadPulse.

Many works employ text mining techniques to distill insightful patterns from thread content. For example, ForumReader [16] and Themail [45] integrate content-parsing algorithms into the exploration of flash forum and email archives, respectively. ForAVis [48] employs sentiment analysis in the search and exploration tasks. Moreover, Convis [25] and MultiConVis [26] combine both sentiment analysis and topic extraction techniques to support the multi-faceted exploration of blog conversations. To meet specific tasks faced by online health communities, VisOHC [30] captures hidden dimensions of threads. Unlike the above approaches, T-Cal first borrows an advanced natural language processing technique [22] to disentangle interleaving conversations. Then, it employs sentiment analysis and keyword extraction techniques to analysis conversational threads from different aspects.

Another research direction is to understand social interactions in conversations. Some work utilizes a radial tree layout to explore user interactions and subgroup formulation in forums [27, 36]. Conversation Map [40] employs a node-link diagram to present social connections in the context of very large-scale conversations in email messages. Perer et al. [37] showed rhythms of relationships using line charts. Revealing interaction and collaboration patterns among team members is one focus of T-Cal. More importantly, our study also targets understanding general trends of discussions as well as analytics and summarization of individual threads.

Visualization of Non-threaded Conversation data

In contrast to messages organized in threads, *non-threaded* conversation data does not have explicit reply relationships. All the messages are usually presented and organized sequentially by their timestamps. Example corpus is transcribed spoken conversations, data in instant chatting clients, and messages in team messaging platforms.

Visualization techniques applied to non-threaded conversations are under-exploited in the literature. Several chat-based systems are designed to show visual feedback on the content of conversations, for example, Conversation Clusters [9], Chat Circles [17], and GroupMeter [31], which encourages social interaction and expressive communication. In a specific application, ArgVis [28] employs interactive graphs to reveal structures of argumentation. Techniques have also been developed to explore verbatim conversation transcripts. For example, ConToVi [20] demonstrates the dynamics of a conversation over time, and NEREx [21] helps discover the relations of named-entity pairs.

However, the above techniques all target at specific applications or conversation data types. They are different from analyzing conversation data on team messaging platforms where the data is large in volume and lasts across a long time span. T-Cal addresses these problems with a calendar-based design to visualize communication patterns at various time scales, including year, month, week, and day, each presenting a different aspect of the conversation data.

Visualization of Temporal Text Data

Our work is also related to techniques for visualizing temporal text corpora. A major approach is based on a visual river metaphor for showing temporal trends in textual data. For example, one of the first systems is ThemeRiver [24]. Similarly, TIARA [51] and ParallelTopics [18] incorporate the LDA algorithm [11] into the river-based design to reveal temporal changes of topics. TextFlow [14] extends TIARA to a more powerful visualization for analyzing topic evolution, critical events, and keyword correlation. More advanced analytics have been applied and integrated with the visual river metaphor, for example, topic competition and collaboration [54, 43]. Some other works employ hierarchical topic modeling to detect topics and reveal their evolution, such as RoseRiver [15], HierarchicalTopics [19], and Liu et al.'s system [32]. In addition to topics, some systems focus on analyzing other aspects of temporal text data, such as diffusion patterns [53], sentiment divergence [13] and anomalous information [56].

The design of T-Cal has been inspired by many of the above systems in presenting temporal patterns of textual data. However, most of the previous works do not address the interconnections among users, messages, and threads, which is essential in our scenario due to the nature of communication. In T-Cal, the combination of information from the three aspects is seamlessly presented with the multi-scale calendar-based visualization and the novel design of ThreadPulse.

DESIGN PROCESS

We followed a typical iterative user-centered design process [33] to develop T-Cal. The entire process can be categorized into three stages as described in the following.

In the first stage, we conducted a formal interview study to understand existing problems and challenges in analyzing conversation data from team messaging platforms, such as Slack [3], Microsoft Teams [1], and Facebook Workplace [4]. From the study results, we derived a set of design requirements to guide our development of T-Cal.

For the second stage, we designed and implemented T-Cal in an iterative manner by involving end users. We chose to use the conversation data generated by a research team using Slack in an IT company. The iterative development process involved several informal discussions with our users in the team, and a poster session within the company that was open to all employees. We refined our prototypes based on the feedback obtained from the users and poster audience.

Lastly, we conducted two case studies to evaluate the effectiveness and usefulness of T-Cal, which will be described later. One case study was related to analyzing team communication patterns from the conversation data of the industry research lab that we closely worked with during the second stage. The other case study was the application of T-Cal for investigating students' behaviors based on Slack group chat logs in a MOOC. The second case study serves an evaluation of T-Cal outside of the application domain in which it was developed.

DESIGN REQUIREMENTS GATHERING

In this section, we describe the first stage of our iterative user-center design process in detail.

Interview Study

In order to understand how people currently explore Slack (or similar tools) and what challenges or opportunities to improve the experience exist, we interviewed six individuals who were frequent users of team messaging platforms. We recruited individuals through reaching out to personal contacts and from members of public online Slack groups or mailing lists. As we were interested in understanding the experiences of people from a wide range of job roles, our interview sample consisted of three management or human resource (HR) professionals who worked for a software company, financial news website, and an architecture firm as well as three people in the software industry, i.e., one data scientist, and two UX designers. Comments from the interviewees contained some common themes with regards to typical uses of the system and needs therein. Interviews lasted approximately 30 minutes and interviewees received a $15 gift card for their participation.

The interviews covered a variety of topics, starting with a general question about how people use and explore Slack conversations and why, for example, *"What do you currently explore Slack for?"* We also focused on information seeking behaviors including media search and archiving needs, and management strategies for dealing with a large volume of information. One example question was *"Can you share any recent examples of times when you have needed to go back and find something?"* Finally, through eliciting specific examples, the interviews touched upon challenges or areas in which Slack lacked functionalities that would be helpful to help people complete their work. For example, we would like to know how they keep up with conversations in Slack. The interviewer took detailed notes during the interviews, which were also recorded and referred back to later for analysis.

Design Requirements

Based on the interview results, we distilled the following design requirements.

R1: Disentangle interleaving conversations. One common issue that came up among both the manager/HR interviewees, the data scientist, and designers was the desire to disentangle interleaving conversational threads in channel discussions. Although Slack had introduced a highly-requested feature, i.e., threaded conversations, in January 2017 prior to the interviews, we found that across all six interviewees, the message threading was not being used. Reasons for this including confusion about where replies to messages could be found (in a separate pane), a feeling that starting a new thread required extra *"clicks"* and was more burdensome than just typing in the general message box, and a general sense that it *"displaces the chronological flow of chat."* Although participants found it hard to read through and tried to interpret conversations quickly from the chronological and one-size-fits-all chat window, as mentioned above, they continued to glean information from the chat this way. Hence, threading in its current state did not help to resolve the issue of separating out different topics and conversations. As a result, the disentanglement of interleaving conversations acts as an entry point for users to explore and analyze conversation data.

R2: Summarize and analyze conversations from different aspects. The three interviewees in higher level roles (manager, HR professionals) had a similar, high-level task of needing to get a sense of what conversational topics were being discussed across the larger team or company as a whole. All three were interested in some form of sentiment analysis that would allow them to know what people were happy about, and more importantly, what problems and issues were cropping up. With the current design of Slack, they were doing this informally. For example, the software manager would browse channels manually to attempt to assess things like what people liked or disliked about the tools they were using. Through similar browsing behavior, the HR would like to know how the business overall was doing, what people were excited about, and what teams were communicating with each other.

R3: Reveal temporal patterns of team communication. One HR professional also mentioned the desire to gain a better understanding of temporal patterns of use among employees, particularly because the team was geographically distributed. She referred to the concept of *"Slacklash"* which was a reaction against Slack's tendency to send notifications at all hours and keep people returning to check it after work hours. She expressed a desire to see how late at night and early in the morning people were working, and working states at weekends, in order to make sure work-life balance issues were not becoming problematic.

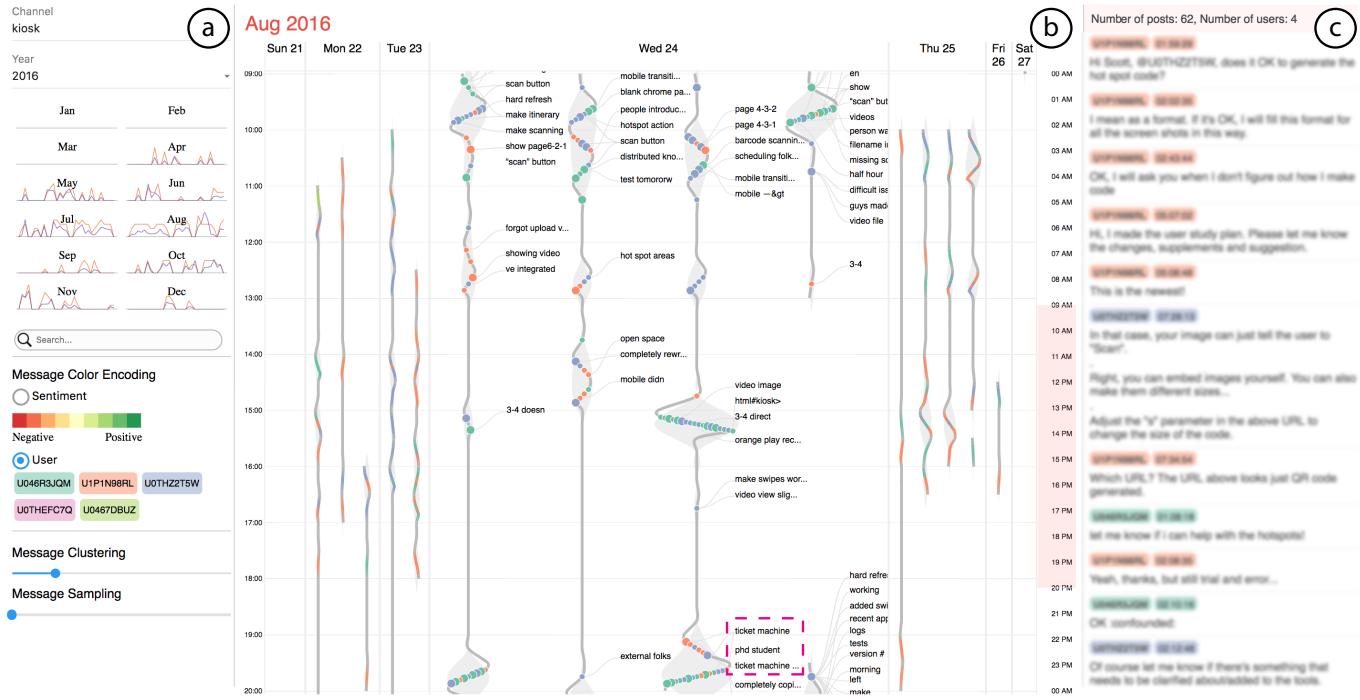

Figure 1. Using T-Cal to explore conversation data generated by team messaging platforms (e.g., Slack). (a) A Control Panel provides configurations for the entire system and entrances to view different datasets with different years and months. (b) A Content panel illustrates team conversational patterns at different time scales, including year, month, and week views. Here, the week of August 21, 2016 is shown with conversational threads presented by a novel visual design called ThreadPulse. (c) A Detail Panel allows users to browse raw messages of selected objects.

R4: Support advanced searching and filtering. Searching and filtering are basic functionalities in data analysis. However, there are challenges in finding information with Slack. For example, one HR professional said, *"You might not always remember what a file was called. However, you have to do your best to remember what it was called or else keep on scrolling and scrolling."* Similarly, one UX designer reported one tedious task in searching. That is, when they did not label an image correctly and were looking for that image. Searching information sent by a specific person and content might help in these scenarios.

T-Cal SYSTEM

Guided by the aforementioned design requirements, we designed and developed T-Cal, a visual analytics system integrating a series of analytical methods with a multi-scale visual interface for analyzing team conversation data generated by Slack (Figure 1). In the following, we first provide an overview of T-Cal, and then describe its visual interface in details.

T-Cal Overview

Inspired by Wijk and Selow's work [52], we borrow a user-familiar interface, i.e., calendar, to intuitively present conversation dataset at multiple time scales. Figure 2 shows an overview of T-Cal architecture, indicating the relationships between the front-end calendar-based visualization, and the back-end data storage, processing, and analytics.

From the database storing the conversation data, the front-end visualization initiates queries to retrieve the information of users and messages within a certain timespan, which could be a year, a month, or a week. The queries are specified by

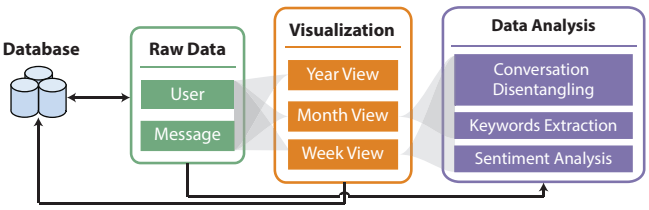

Figure 2. Overview of T-Cal architecture.

user interactions from the visualization. Based on the retrieved users and messages, three analytical processes are performed, including 1) a conversation disentangling method [46] to split interleaving conversations into threads, 2) a keyword extraction algorithm called RAKE [39] to distill key information for each thread, and 3) a sentiment analysis method [10] to infer sentiment of each message. All the original data (i.e., users and messages) and the derived information (i.e., threads, keywords, and sentiments) are provided to the visualization as needed in views of different time scales (i.e., year, month, and week views). Each view may request different information (e.g., the year view only demands user and message data).

Particularly, the front-end T-Cal visual interface consists of three panels (Figure 1): a) a Control Panel providing configurations for the entire system (**R4**), b) a Content Panel displaying the data at different levels of details (including year (**R3**), month (**R1**, **R2**), and week views (**R1**, **R2**, **R3**), and c) a Detail Panel showing the raw messages for users to examine specific conversations. In the rest of this section, we describe the detailed visual design of each view.

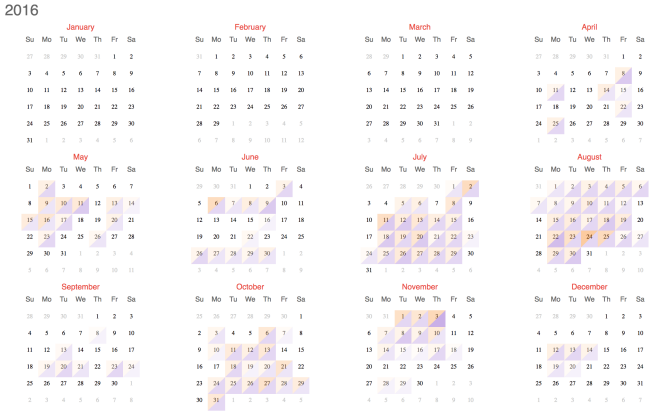

Figure 3. The Year View presents overall activities of one team in a year, using heatmaps of two quantities: the volume of messages (in orange), and the number of participating team members (in purple).

Summarizing Yearly Activities

The Year View (Figure 3) is aimed at presenting overall activities of a team in one year (**R3**). Users are able to switch between different years and teams with the Control Panel. Akin to a normal calendar, the Year View consists of the twelve months, using heatmaps to reflect daily team activities. Based on a design in [5], we divide each day cell along the diagonal into two triangles to encode two important quantities: the volume of messages (in orange), and the number of participating team members (in purple). Color density is mapped to the value of these two quantities (the darker the higher). For example, Figure 3 shows that the team formed in April 2016 and had a lot of activities in July and August as indicated by the darker purple and orange color. Although this implementation involves the above two quantities, other pairs of team activity indicators can be easily integrated upon a user's preference.

Exploring Monthly Aggregations

A user can easily dive into a month from the Year View in the Content Panel (Figure 4). The Month View unfolds the relationships among team members, messages, and conversational threads (**R1**, **R2**). It can be used to answer analytical questions such as *"How many threads were there every day," "Who were the most active members of the team,"* and *"Which team members were frequently talking together?"*

In this view, team members are presented as circles with the color indicating user identity and the size indicating the number of messages created by them. Team members involved in the same thread are packed together in a bigger hollow circle using the circle packing algorithm [47]. When a user hovers over a thread, a word cloud is displayed to help grasp the key information of the thread. Semantically meaningful keywords are extracted and selected based on RAKE [39]. The color corresponds team member identity and the size indicates keyword weight. For example, we can observe that words of highest importance such as `updated` and `related` were mentioned by one team member (green) on August 19, and he/she seemed to be very active every day based on the size and the number of the green circles across the whole month.

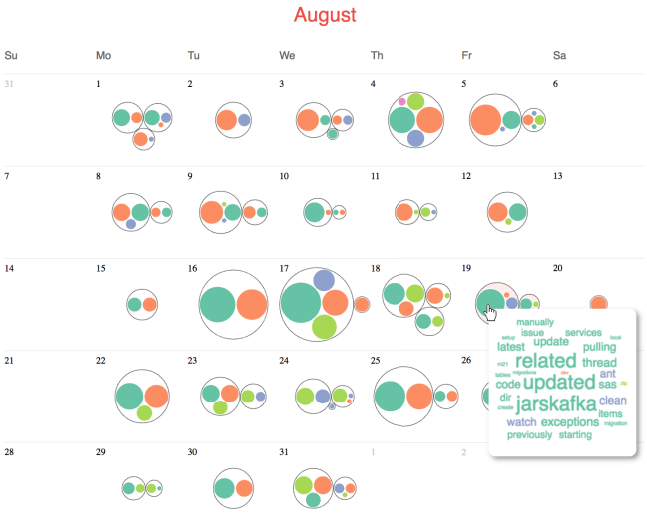

Figure 4. The Month View shows relationships among team members, messages, and conversational threads. A word cloud is initiated when a user hovers over a thread to summarize the key information.

The above color-coding of user identity, however, is not scalable to channels with a large number of users. One possible solution is to represent the top active users (e.g., top ten) with colors and all the other users with a default color such as gray. The goal here is to emphasize key players of a channel because not all users are of the same importance for analysis.

Inspecting Weekly Conversations

To retrieve more details, a user can shift the exploration from aggregations in the Month View to temporal dynamics and contextual details of each conversational thread (**R1**, **R2**) in the Week View (Figure 1(b)). As the intensity of team discussion varies largely across time, understanding temporal patterns of conversational threads is a critical analytical task (**R3**). Thus, this view aims to answer users' questions like *"When did the most intense discussion happen, and how did it go along time," "How team members were interacting with each other in a conversation, such as who followed/replied whom,"* and *"What were people mainly talking about?"*

To address these issues, we develop a novel visual design, ThreadPulse, to present individual conversational threads in the Week View, inspired by the visual metaphor of an electrocardiogram to reveal the "pulses" of a conversation. Following the "Overview first, zoom and filter, details on demand" mantra [41], ThreadPulse offers two different levels of details for visualizing a conversation, coupled with user interaction enabled in the Week View. In particular, as shown in Figure 1(b), by default the Week View displays conversational threads on each day with a *coarse-level* of ThreadPulse (e.g., Tuesday, August 23; also see Figure 5(d)), and a specific day can be expanded to reveal more details about the threads with a *fine-level* of ThreadPulse (e.g., Wednesday, August 24; also see Figure 5(c)). Moreover, a user can initiate a time window, indicated as a red box on the right of the Week View, to browse data at different timespans. This interaction also affects the appearance of ThreadPulse. Details about the design of ThreadPulse will be discussed below.

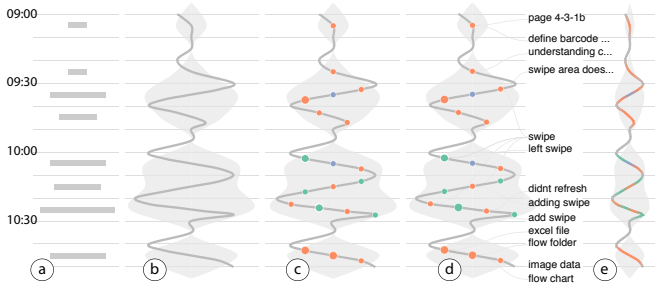

Figure 5. The process of generating ThreadPulse. (a) An adaptive binning approach divides the time frame into multiple N-minute time slots (gray bars indicate message intensity values). (b) A volume chart and a zig-zag curve is drawn to present message intensity across time. (c) Messages are depicted as circles along the curve. (d) Keywords of threads are placed on the side. (e) The coarse-level design of ThreadPulse hides details such as keywords and the number of messages in each time slot.

ThreadPulse Design

The goal of ThreadPulse is to simultaneously reveal overall temporal trends of a conversational thread, as well as informative details of messages such as team members who created them, message content, and sentiment. Also, we aim to address the issues of wide ranges in message intensity and scalability. The visual design of the fine-level ThreadPulse consists of three subcomponents: 1) a background with a volume chart and a thread curve, 2) a foreground with individual message data, and 3) a side-area with keywords. The coarse-level ThreadPulse only contains the first two subcomponents.

Step 1. To generate the background volume chart and curve, we first apply an adaptive binning approach to divide messages into equal time slots (Figure 5(a)). The time length (e.g., in minutes) of time slots is determined dynamically based on the available screen real estate. For example, if a user zooms in (i.e., having more pixels) by brushing on the right of the Week View, the number of minutes represented by each time slot decreases, by keeping the pixel length of time slots as constant at different zoom levels. After this, we draw a volume chart, and construct a smooth curve along time in a zig-zag manner (Figure 5(b)). The amplitude of the volume chart and the curve is mapped to the message intensity across time. We choose the curve design because it conveys the temporal continuity of a thread better than a traditional histogram.

Step 2. To encode individual message data, we further draw messages on top of the background. Messages are positioned by the intersections between their timestamps and the curve. In the fine-level design, each message is shown as a circle with the color representing user identity or other quantities such as sentiment (Figure 5(c)). In the coarse-level design, the corresponding curve segment is color-coded directly (Figure 5(e)).

In some cases, it is not possible to draw every message on the curve when the number of messages is large in a time slot. We employ message *clustering* and *sampling* in order to reduce visual clutter. With instant chatting systems, users usually post multiple quick messages in a short time T, for example, *"Sure!"* immediately followed by another *"Will do."* We group these messages created by the same team member, and use bigger circles to indicate such aggregated messages (with the size corresponding to message number). Further, we rank the messages within each time slot using RAKE [39], and only show messages in the top-N of the ranked list when there is not enough space to present all messages within that time slot. The above parameters, T for clustering and N for sampling, can be dynamically configured by a user on the Control Panel.

Step 3. For the fine-level ThreadPulse, we also show the extracted RAKE keywords of messages on the side (Figure 5(d)). The vertical locations of the keywords are determined by a force-directed layout method [8] to avoid collisions. Links are drawn between keywords and corresponding messages on the curve, which is a many-to-many relationship. This allows users to discover important keywords and messages by observing the links.

Arrangement of multiple threads. Once the ThreadPulse representation is generated for each conversational thread of one day, we place them on the corresponding day of the Week View based on a greedy layout approach. We sort all the threads by starting time and place them according to that order. If a thread to be placed overlaps with ones that are already drawn, we simply move it to the next column on the right, otherwise we just place it in the same column (Figure 1(b)).

Iterative Design Process

We conducted the design of ThreadPulse through an iterative process by working with our end users.

At first, we explored other design alternatives to present both general temporal trends and detailed message data of a thread. One candidate was the *bee swarm chart*, which closely packs messages in each time slot without overlap (Figure 6(a)). Although the bee swarm chart meets the requirements of showing both overall and detailed information, it fails to reveal the sequence of messages, which is critical for understanding the context of conversations. Another option was the *FluxFlow plot* [56], which depicts the information even more compactly (Figure 6(b)). However, it has the same limitations as the bee swarm chart, and it does not accurately convey message timestamps due to circle packing.

Then, we came up with a curve-based design (Figure 6(c)). Unlike the bee swarm chart and FluxFlow plot, it preserves the continuity and sequence of messages, and encodes the timestamps with vertical positions precisely. To generate the curve, for each time slot, we select two points that the curve needs to pass through, i.e., the top-left and the bottom-right corners. However, this approach produces curves with too many zig-zags. Finally, as shown in Figure 6(d), we reduced the number of zig-zags by half via alternating between the top-left & bottom-right corners and the top-right & bottom-left ones for consecutive time slots (hence there is one point overlapping for every two neighboring slots). This design strikes a balance between being concise and informative, as well as an intuitive visual metaphor for conversations.

During the design process, we received much positive feedback from end users with ThreadPulse, compared with the other two alternatives, for example, *"This design [ThreadPulse] looks better because it shows a lot of information and at the same time not that complicated."* However, some users pointed out that it was not easy to perceive the message

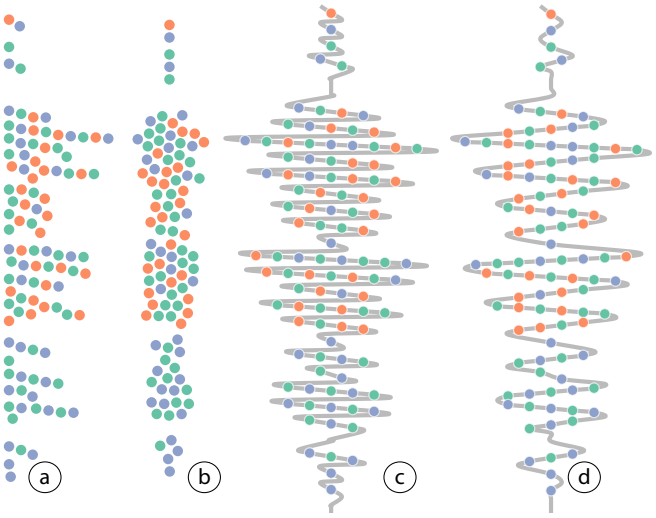

Figure 6. Design alternatives for ThreadPulse showing the same thread during the same time period. (a) A bee swarm chart. (b) The FluxFlow plot. (c) An initial curve-based design. (d) The final curve-based design used in ThreadPulse.

intensity with the amplitude of the curve. They also mentioned that they would like to see a summary of the actual content of threads, in addition to the temporal patterns. Therefore, based on these comments, we further refined our design by adding the volume chart and the keywords (Figure 5(d)).

Retrieving Messages and Manipulating Views

T-Cal provides the Detail Panel to allow users to verify findings and inspect raw messages of interest (Figure 1(c)). The Detail Panel is interactively synchronized with user interaction on other panels. For example, when the Month or Week View is loaded, it shows all messages within a month or a week. If a user selects a conversational thread, the raw messages will update accordingly.

Messages can also be retrieved by searching (e.g., by message keywords or team member meta-data) and filtering (e.g., by users) on the Control Panel (Figure 1(a)), in combination with the main visualization (**R4**). Search results are interactively reflected on the corresponding view in the Content Panel (Figure 1(b)), thus offering a multi-scale search capability. However, more advanced information retrieval techniques should be integrated to fully meet the user requirements. But this is out the scope of this paper. Further, the Control Panel provides an entrance to the Year and Month Views, and allows users to easily switch between different years and months, as well as to select different teams. The Control Panel also configures the visualizations such as message color encoding scheme, and message clustering & sampling parameters.

VALIDATION WITH EXPERTS

In this section, we demonstrate the effectiveness and usefulness of T-Cal using two case studies with experts from different domains. One involved two employees of the industry research lab we closely worked with during the design process. The other involved two teaching staff of a MOOC course at a university, as a validation of T-Cal outside of the domain where it was iteratively designed.

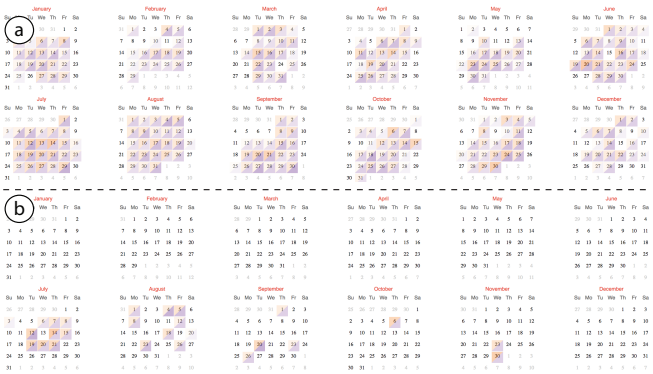

Figure 7. (a) The Year View of the `dev` channel in 2016. (b) The filtered Year View after searching the keyword `formula`.

We conducted semi-structured interviews with our experts, each lasting about one hour. During the interviews, we first introduced the views and features of T-Cal using their own datasets. Then, the experts took control to investigate their data, during which they might ask questions about the system. We instructed them to perform their tasks in a think-aloud protocol. Finally, we discussed with the experts about the insights gained during their exploration, as well as the strengths and weaknesses of T-Cal. We took notes when necessary and recorded the entire sessions for later analysis.

Analyzing Team Communication in the Enterprise

Expert Users and Dataset

In this case study, we contacted two employees of the team with different job roles, including one product manager (PM) who leads an engineering team, and one research scientist (RS) who is involved in a research project.

We approached the two experts independently in different interview sessions. Each of them was interested in a subset of the Slack channels within the research lab. The chat logs of all the channels of interests were captured from June 2015 to May 2017. Specifically, the PM looked into two channels regarding development (17,095 messages) and quality assurance (19,452 messages) of one product, and the RS inspected a channel dedicated to an on-going project (3,153 messages).

Interpreting Enterprise Messaging Data

The PM explored the development (named `dev`) and the quality assurance (named `qa`) channels together, because these two channels were highly related in support to the product that his team worked on. Although T-Cal currently does not support data comparison, he achieved his goal by easily launching two instances of the system with different channels and placing them side by side.

After opening the the Year View 2016 of the `dev` channel, the PM searched for the keyword `formula` on the Control Panel, because it was related to one of the critical product features. The Year View updated accordingly, and he found that this word frequently appeared in August, as observed from the heatmap (Figure 7). The PM said, *"We were intensively working on a feature for a product in July and August 2016."* Thus, he clicked August 2016 on both sides to get more details (Figure 8(a) and (b)). By hovering over a thread on

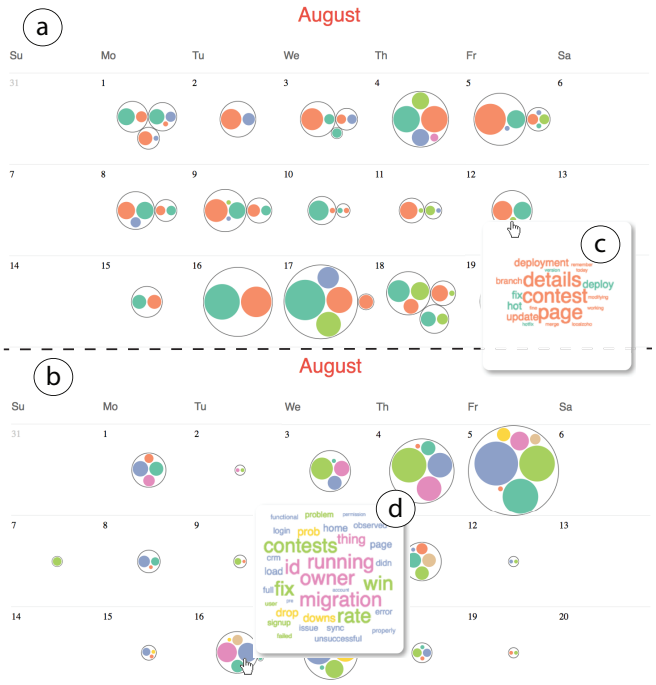

Figure 8. (a)(b) Monthly activities of the dev and qa channels, respectively. (c) The word cloud for August 12 in the dev channel. (d) The word cloud for August 16 in the qa channel.

August 12 in the dev channel, he saw a keyword contest in the word cloud (Figure 8(c)). Further, he saw the same keyword appeared in the word cloud of another thread on August 16 in the qa channel (Figure 8(d)). He commented, *"In August, we were developing the feature by checking out the 'contest' (git branch). We discussed it in the* dev *channel during development, and switched to the* qa *channel after we ship the feature."* Similarly, the PM found several other cases in which an initiation raised in the dev channel and later was moved to the qa channel. He further added, *"The system helps me trace topic transitions across the two channels. We can investigate how a topic or a feature evolved to reflect on the efficiency of developing products."*

The RS was interested in exploring the team channel. She explored the overall team activities of last year with the Year View (Figure 3). She pointed to July 2016 that had darker colors on the heatmap, and commented, *"At the end of June, we went to do field work, so that was probably a lot of discussions. We sent messages back here to update people on what we have done every day"*. Then, November drew her attention, where colors were darker in the first two weeks. She said, *"November was the time of a technology conference, which was another busy time because we were preparing for a talk in the conference"*. She further added, *"I feel that I am connected to the data. The system is useful for doing some reflection on our past activities."*

She also noticed August 2016 when lots of discussions happened, where August 24 caught her eyes because there were many high-volume threads, observed from the sizes of circles. Thus, she entered the Week View and expanded that day to inspect what happened (Figure 1(b)). She found a thread

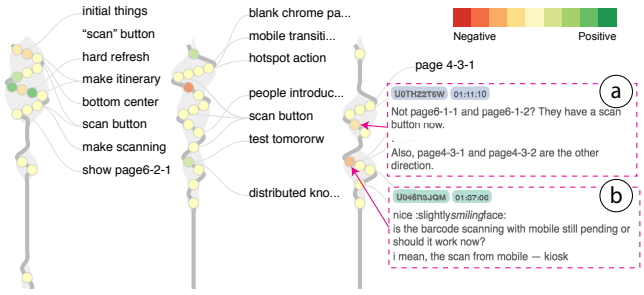

Figure 9. ThreadPulse with the color representing message sentiment. (a)(b) The raw data of two messages in red boxes, respectively.

containing a keyword ticket machine (highlighted in red boxes in Figure 1), and said, *"Showing keywords is a good way of going through a thread. Ticket machine was talked about a lot, because we were discussing whether to include a feature or not. It is useful that the most important stuff is highlighted."* In addition, the RS especially liked the design of ThreadPulse, commenting: *"The curve is intuitive and the information it conveys is easy to understand. I think it is pretty useful and clear."* To discover problems or conflicts that emerged in the conversations, the RS switched the message color encoding to sentiment (Figure 9). She discovered some negative messages aligned along with neutral and positive messages, observed from the color of the circles. She checked each negative messages and found that they discussed one pending function of their project (Figure 9(a) and (b)). Thus, she commented, *"Showing messages by sentiment can help me quickly identify problems we discussed."*

Examining MOOC Study Groups

Expert Users and Dataset

For this case study, we interviewed two teaching staff at the same time: one university professor (course instructor) and one graduate student (teaching assistant), who released a MOOC on Coursera [2] in summer 2017. Over 50 students from 24 unique countries signed up the course. In this course, students were expected to finish a group project. To support distant collaboration on group projects, they created a Slack group for all students and a private channel for each team to facilitate the real-time communication and document sharing.

Chat logs on Slack were captured during the course. At the time of writing, 13 students from four teams had finished all milestones and completed the course, and the final dataset contained 7,682 messages in total. Of particular interest, our experts required a specific type of conversation disentangling by categorizing messages into two main categories: *course-related discussions* and *casual chats*. To achieve this, we employed SeededLDA [38] to disentangle conversations by using the seeded words provided by our experts.

Understanding Students' Communication

The goal of our experts was to understand communication patterns of individual teams and perform comparative analysis among different teams. They tried to understand questions such as *"How do students overcome challenges such as freeriding and timezone differences in a group project,"* and *"Which factors have effects on students' collaboration?"* From the Year View of 2017 for different teams, they observed that all of

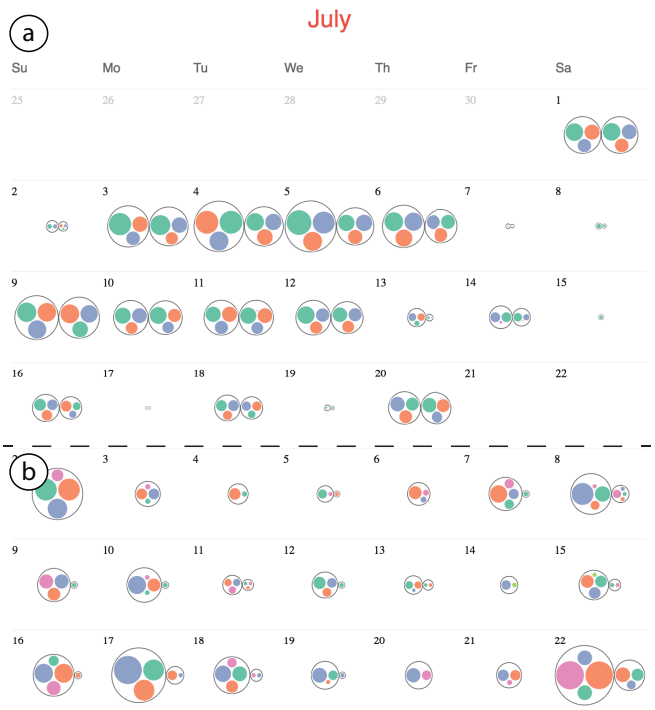

Figure 10. Monthly activities of `Team1` and `Team2` in July 2017. On each day, the two circles represent a course-related thread (left) and a casual-chat thread (right). (a) Team1 created course-related messages and chat with similar volume every day. (b) Conversations in Team2 were mainly course-related.

them started the project at the end of June and ended in early August. The instructor first selected `Team1` and focused on activities in July with the Month View (Figure 10(a)). Overall, almost all course-related and chatting threads involved the three team members, represented as blue, orange, and green circles. The experts observed that each circle was of similar size, indicating that the team members contributed evenly in their group discussions. The teaching assistant noted, *"We did not receive any reports of freeriding in the weekly peer evaluations during the course period. This diagram confirms our knowledge with an intuitive presentation."*

Then the experts shifted the focus to the temporal trends of each thread. The instructor selected one week (the week of July 9, 2017) with many discussions to show it in the Week View (Figure 11(a)). They found that almost all threads had the highest conversation intensity during 15 : 00 and 18 : 00, and few messages were created outside that time period, as indicated by the amplitude of ThreadPulse. The instructor said, *"Due to time zone difference, this team seemed to develop appropriate strategies to cope with timezone issues in the collaboration."* She added, *"It seems this team worked in a synchronous way. I mean, all team members had a consensus on the available time at the beginning of the project, and they fixed the time for communication."*

Moreover, our experts were interested in comparing communication patterns between different teams. Hence, they switched to `Team2` and also inspected their activities in July (Figure 10(b)). The instructor observed that some days only had one thread, indicating that students either discussed

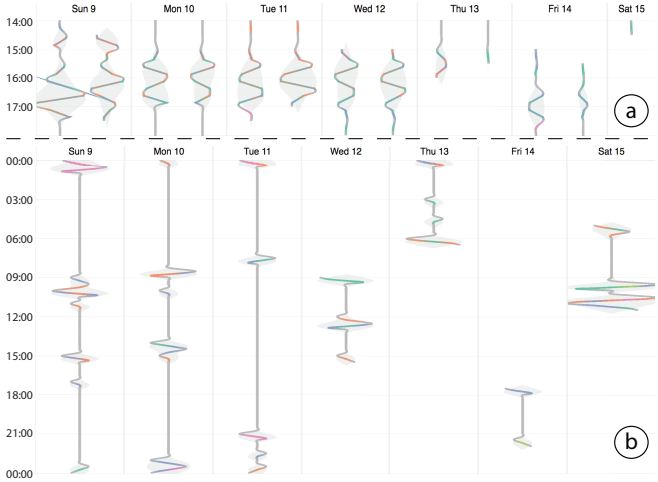

Figure 11. Comparison of temporal patterns of conversational threads between `Team1` and `Team2` in July. (a) `Team1` worked in a synchronous manner, (b) whereas `Team2` collaborated asynchronously.

course-related content or just chatted together. For the days with two threads, the message volumes of these threads were quite different. By examining the content in the Detail Panel, she found that most threads, especially represented with bigger circles, were course-related, commenting: *"Compared to `Team1`, it is interesting that they have distinct conversation styles. The pattern drives us to find the reason behind it."*

To further compare the temporal patterns of conversations of this team to `Team1`, the instructor entered one week in July (Figure 11(b)). With the Week View, she noticed that curve segments with greater amplitude distributed across the day, meaning that conversations happened throughout the day. *"Oh, this team applied asynchronous strategy to collaboration,"* she explained, *"Each team member worked on the project on their own schedule. So responses would be deferred until other team members came online. The team did not tend to involve every member in the communication at the same time, but only those who were available."*

GENERAL FEEDBACK OF EXPERTS AND DISCUSSION

During our interviews, all experts showed great interests in using T-Cal, and appreciated the insights found in their data. Especially, the MOOC instructor commented, *"These stories allow us to understand the strategies teams used to overcome the challenges of remote collaboration when working in the MOOC. We could use this information to provide design implications for future MOOC instructors and students."* Although T-Cal was shown to be effective, we still identified several limitations and issues from the interviews. Here we report them and discuss possible solutions.

Support comparative analysis of multiple channels. The PM and two MOOC experts performed a similar high-level task of monitoring and comparing conversations across multiple channels. To be specific, the PM wanted to keep track of several related channels for developing one product and the MOOC experts desired to compare communication patterns among multiple teams. Currently, they have to launch two T-Cal applications and align them side by side to perform the analysis. We plan to address this requirement using two

approaches. First, datasets from multiple channels can be merged and encoded using the design in [5] for comparison. Second, the presentation and interactive linking between two channels can be supported with a side-by-side visualization.

Employ better approaches for conversation disentanglement. Both the RM and RS commented that the techniques for disentangling conversations should be improved. When observing the Week View shown in Figure 1(b), the RS noted, *"Does it imply that we were having four conversations at the same time on August 24? It looks strange to me."* The problem of conversation disentanglement is challenging due to the complexity and variation of human languages [22]. Although the analytical results are not satisfactory in some cases, the limitation may be resolved when advanced natural language processing techniques are proposed in the future. Also, the T-Cal visualization is designed to work with any conversation disentangling algorithm that is not our focus in this paper.

Facilitate the analysis of team collaboration. The RS pointed out that the Month View could effectively reveal conversational threads but not team collaborations (e.g., a social network of team members). She commented, *"I feel some information about people is lost [in the Month View]. If I am concerned about whether some people are constantly talking with each other, I need these people to be grouped to show that they are forming some units. But currently, they are grouped by threads."* To facilitate this analysis, T-Cal can be augmented with node-link diagrams of team collaborations on each day cell in the Month View. Users can switch between the two representations based on their analytical goals.

Assist event-based exploration with context. The RS in the first case study observed that a large number of messages were created on August 24, 2016 (Figure 3), but she did not remember what happened at that time. *"It would be helpful if T-Cal can link the Year View with my personal event calendar, because this is a way to understand the context and get a sense of what was going on,"* she commented. Although not supported currently, we argue that this feature can be easily added with available calendar services. Further, in the Week View, the RS wanted to conduct event-based analysis, such as *"when did the team members start to discuss one problem and when was the problem solved?"* To achieve this, some event detection algorithms (e.g., [7]) can be employed, and the detected events could be encoded visually in the views, for example, using colored text labels as in Google Calendar.

Support conversational threads lasting across days. Experts' feedback from the case studies implies that ThreadPulse is intuitive and informative for presenting conversational threads with both overall trends and detailed data. However, with the current calendar-based design of T-Cal, ThreadPulse cannot visualize threads across day boundaries. All experts noted this issue during our interviews. But considering ThreadPulse alone, it is applicable for showing conversations with any time length. For the current Week View of T-Cal, an easy way to accommodate multi-day threads is to employ user interaction for linking related threads on consecutive days.

STUDY LIMITATIONS AND GENERALIZATION

One limitation of our design study is the sample size, as we only involved six participants in the requirement gathering and four experts in the evaluation based on a qualitative method. While they are from different domains and in various job roles, thus covering a range of our end users, a long-term deployment study is needed to further collect users' feedback. Such a study will allow us to assess interface efficacy and discover deeper insights about the usage of T-Cal in real-life settings [42].

It is worth noting that although T-Cal is designed for visual analytics of team messaging platforms, the system can be applied to visualizing other online communication systems, such as emails, forums, and blog posts. Many aspects of T-Cal, such as the calendar-based approach for multi-scale visual exploration, remain useful for general applications. For example, the Year View can display a heatmap of other factors of interests in the particular application domain, or employ other visual designs, such as glyphs for showing network patterns of asymmetric relations [12]. The Month View can be augmented with node-link diagrams for visualizing dynamic social networks of members varying over days.

In addition, the ThreadPulse can be extended to visualize the temporal distribution and details of datasets containing items with temporal information. For example, in event sequence data analysis, ThreadPulse can be applied to visualize general trends of a sequence as well as detailed information of individual events, such as event type. To be specific, one circle represents one event, and event type can be encoded with color. The keywords can then depict the contextual information of an event sequence.

CONCLUSION AND FUTURE WORK

We have presented T-Cal, an interactive visualization for assisting the analysis of team communication and collaboration. T-Cal displays the conversation data captured in team messaging platforms in multiple scales with a calendar-based visualization. We have also introduced ThreadPulse for showing conversational threads in multiple levels of detail. By involving expert users, we derived a set of design requirements and iteratively refined the system in a three-stage design process. For evaluation, we reported two case studies to illustrate the effectiveness and usefulness of T-Cal with realistic datasets in two different applications.

In the future, we plan to pursue the comparative analysis of conversation data from two teams. Moreover, we want to focus on the event-based analysis in data, e.g., understanding when a problem was proposed, how it evolved, and when it was solved, by integrating other sources of data. Finally, we plan to further evaluate T-Cal in different application domains and conduct a long-term deployment study.

ACKNOWLEDGMENTS

We thank all the experts and users in our studies, as well as Dr. Huamin Qu and Dr. Daniel Avrahami for their constructive feedback. This work is partially supported by FXPAL as an internship project, University of Minnesota Start-up Funding, and National 973 Program of China (2014CB340304). Jian Zhao is the corresponding author.

REFERENCES

1. 2017. Microsoft Teams—Group Chat Software. https://products.office.com/en-us/microsoft-teams/group-chat-software. (2017). Accessed: 2017-8-18.
2. 2017. Online Courses From Top Universities. Join For Free. https://www.coursera.org/. (2017). Accessed: 2017-8-18.
3. 2017. Slack: Where work happens. https://www.slack.com/. (2017). Accessed: 2017-8-18.
4. 2017. Workplace by Facebook - A New Way to WorkâĂŐ. https://www.facebook.com/workplace. (2017). Accessed: 2017-8-18.
5. Basak Alper, Benjamin Bach, Nathalie Henry Riche, Tobias Isenberg, and Jean-Daniel Fekete. 2013. Weighted Graph Comparison Techniques for Brain Connectivity Analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 483–492. DOI: http://dx.doi.org/10.1145/2470654.2470724
6. Henrik Artman and Christer Garbis. 1998. Team communication and coordination as distributed cognition. In *9th Conference of Cognitive Ergonomics*. 151–156.
7. Farzindar Atefeh and Wael Khreich. 2015. A Survey of Techniques for Event Detection in Twitter. *Comput. Intell.* 31, 1 (Feb. 2015), 132–164. DOI: http://dx.doi.org/10.1111/coin.12017
8. Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G Tollis. 1998. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR.
9. Tony Bergstrom and Karrie Karahalios. 2009. Conversation Clusters: Grouping Conversation Topics Through Human-computer Dialog. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 2349–2352. DOI: http://dx.doi.org/10.1145/1518701.1519060
10. Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python*. O'Reilly Media, Inc.
11. David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.
12. Ulrik Brandes and Bobo Nick. 2011. Asymmetric Relations in Longitudinal Social Networks. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (2011), 2283–2290. DOI: http://dx.doi.org/10.1109/TVCG.2011.169
13. Nan Cao, Lu Lu, Yu-Ru Lin, Fei Wang, and Zhen Wen. 2015. SocialHelix: visual analysis of sentiment divergence in social media. *Journal of Visualization* 18, 2 (01 May 2015), 221–235. DOI: http://dx.doi.org/10.1007/s12650-014-0246-x
14. W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong. 2011. TextFlow: Towards Better Understanding of Evolving Topics in Text. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (Dec 2011), 2412–2421. DOI: http://dx.doi.org/10.1109/TVCG.2011.239
15. Weiwei Cui, Shixia Liu, Zhuofeng Wu, and Hao Wei. 2014. How Hierarchical Topics Evolve in Large Text Corpora. *Visualization and Computer Graphics, IEEE Transactions on* 20, 12 (2014), 2281–2290. DOI: http://dx.doi.org/10.1109/TVCG.2014.2346433
16. Kushal Dave, Martin Wattenberg, and Michael Muller. 2004. Flash Forums and forumReader: Navigating a New Kind of Large-scale Online Discussion. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*. 232–241. DOI: http://dx.doi.org/10.1145/1031607.1031644
17. Judith Donath and Fernanda B. Viégas. 2002. The Chat Circles Series: Explorations in Designing Abstract Graphical Communication Interfaces. In *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques (DIS '02)*. ACM, New York, NY, USA, 359–369. DOI: http://dx.doi.org/10.1145/778712.778764
18. W. Dou, X. Wang, R. Chang, and W. Ribarsky. 2011. ParallelTopics: A probabilistic approach to exploring document collections. In *2011 IEEE Conference on Visual Analytics Science and Technology (VAST)*. 231–240. DOI: http://dx.doi.org/10.1109/VAST.2011.6102461
19. W. Dou, L. Yu, X. Wang, Z. Ma, and W. Ribarsky. 2013. HierarchicalTopics: Visually Exploring Large Text Collections Using Topic Hierarchies. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (Dec 2013), 2002–2011. DOI: http://dx.doi.org/10.1109/TVCG.2013.162
20. Mennatallah El-Assady, Valentin Gold, Carmela Acevedo, Christopher Collins, and Daniel Keim. 2016. ConToVi: Multi-Party Conversation Exploration using Topic-Space Views. *Computer Graphics Forum* 35, 3 (2016), 431–440. DOI: http://dx.doi.org/10.1111/cgf.12919
21. Mennatallah El-Assady, Rita Sevastjanova, Bela Gipp, Daniel Keim, and Christopher Collins. 2017. NEREx: Named-Entity Relationship Exploration in Multi-Party Conversations. *Computer Graphics Forum* 36, 3 (2017), 213–225. DOI:http://dx.doi.org/10.1111/cgf.13181
22. Micha Elsner and Eugene Charniak. 2010. Disentangling Chat. *Comput. Linguist.* 36, 3 (Sept. 2010), 389–409. DOI:http://dx.doi.org/10.1162/coli_a_00003
23. S. Fu, J. Zhao, W. Cui, and H. Qu. 2017. Visual Analysis of MOOC Forums with iForum. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (Jan 2017), 201–210. DOI: http://dx.doi.org/10.1109/TVCG.2016.2598444

24. S. Havre, E. Hetzler, P. Whitney, and L. Nowell. 2002. ThemeRiver: visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics* 8, 1 (Jan 2002), 9–20. DOI:http://dx.doi.org/10.1109/2945.981848

25. E. Hoque and G. Carenini. 2014. ConVis: A Visual Text Analytic System for Exploring Blog Conversations. *Computer Graphics Forum* 33, 3 (2014), 221–230. DOI: http://dx.doi.org/10.1111/cgf.12378

26. Enamul Hoque and Giuseppe Carenini. 2016. MultiConVis: A Visual Text Analytics System for Exploring a Collection of Online Conversations. In *Proceedings of the 21st International Conference on Intelligent User Interfaces (IUI '16)*. ACM, New York, NY, USA, 96–107. DOI: http://dx.doi.org/10.1145/2856767.2856782

27. Sujana Jyothi, Claire McAvinia, and John Keating. 2012. A Visualisation Tool to Aid Exploration of Students' Interactions in Asynchronous Online Communication. *Comput. Educ.* 58, 1 (2012), 30–42. DOI: http://dx.doi.org/10.1016/j.compedu.2011.08.026

28. Areti Karamanou, Nikolaos Loutas, and Konstantinos Tarabanis. 2011. *ArgVis: Structuring Political Deliberations Using Innovative Visualisation Technologies*. Springer Berlin Heidelberg, Berlin, Heidelberg, 87–98. DOI: http://dx.doi.org/10.1007/978-3-642-23333-3_8

29. B. Kerr. 2003. Thread Arcs: an email thread visualization. In *Information Visualization, 2003. INFOVIS 2003. IEEE Symposium on*. 211–218. DOI: http://dx.doi.org/10.1109/INFVIS.2003.1249028

30. B.C. Kwon, S. Kim, S. Lee, J. Choo, J. Huh, and J.S. Yi. 2016. VisOHC: Designing Visual Analytics for Online Health Communities. *Visualization and Computer Graphics, IEEE Transactions on* 22, 1 (2016), 71–80. DOI:http://dx.doi.org/10.1109/TVCG.2015.2467555

31. Gilly Leshed, Diego Perez, Jeffrey T. Hancock, Dan Cosley, Jeremy Birnholtz, Soyoung Lee, Poppy L. McLeod, and Geri Gay. 2009. Visualizing Real-time Language-based Feedback on Teamwork Behavior in Computer-mediated Groups. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 537–546. DOI:http://dx.doi.org/10.1145/1518701.1518784

32. S. Liu, J. Yin, X. Wang, W. Cui, K. Cao, and J. Pei. 2016. Online Visual Analytics of Text Streams. *IEEE Transactions on Visualization and Computer Graphics* 22, 11 (Nov 2016), 2451–2466. DOI: http://dx.doi.org/10.1109/TVCG.2015.2509990

33. Tamara Munzner. 2009. A Nested Model for Visualization Design and Validation. *IEEE Transactions on Visualization and Computer Graphics* 15, 6 (Nov. 2009), 921–928. DOI: http://dx.doi.org/10.1109/TVCG.2009.111

34. S. Narayan and C. Cheshire. 2010. Not Too Long to Read: The tldr Interface for Exploring and Navigating Large-Scale Discussion Spaces. In *System Sciences, 2010. Proceedings of the 43rd Annual Hawaii International Conference on*. 1–10. DOI: http://dx.doi.org/10.1109/HICSS.2010.288

35. Paula S. Newman. 2002. Exploring Discussion Lists: Steps and Directions. In *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '02)*. 126–134. DOI: http://dx.doi.org/10.1145/544220.544245

36. V. Pascual-Cid and A. Kaltenbrunner. 2009. Exploring Asynchronous Online Discussions through Hierarchical Visualisation. In *Information Visualization, 2009 13th International Conference*. 191–196. DOI: http://dx.doi.org/10.1109/IV.2009.14

37. Adam Perer, Ben Shneiderman, and Douglas W. Oard. 2006. Using rhythms of relationships to understand e-mail archives. *Journal of the American Society for Information Science and Technology* 57, 14 (2006), 1936–1948. DOI:http://dx.doi.org/10.1002/asi.20387

38. Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daume III, and Lise Getoor. 2014. Understanding MOOC Discussion Forums using Seeded LDA. In *9th ACL Workshop on Innovative Use of NLP for Building Educational Applications*.

39. Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extraction from Individual Documents*. John Wiley & Sons, Ltd, 1–20. DOI:http://dx.doi.org/10.1002/9780470689646.ch1

40. Warren Sack. 2000. Discourse diagrams: Interface design for very large-scale conversations. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*. IEEE, 10.

41. B. Shneiderman. 1996. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE Symposium on Visual Languages*. IEEE. DOI:http://dx.doi.org/10.1109/VL.1996.545307

42. Ben Shneiderman and Catherine Plaisant. 2006. Strategies for Evaluating Information Visualization Tools: Multi-dimensional In-depth Long-term Case Studies. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization (BELIV '06)*. ACM, New York, NY, USA, 1–7. DOI:http://dx.doi.org/10.1145/1168149.1168158

43. G. Sun, Y. Wu, S. Liu, T. Q. Peng, J. J. H. Zhu, and R. Liang. 2014. EvoRiver: Visual Analysis of Topic Coopetition on Social Media. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (Dec 2014), 1753–1762. DOI: http://dx.doi.org/10.1109/TVCG.2014.2346919

44. Gina Danielle Venolia and Carman Neustaedter. 2003. Understanding Sequence and Reply Relationships Within Email Conversations: A Mixed-model Visualization. In

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03). 361–368. DOI: http://dx.doi.org/10.1145/642611.642674

45. Fernanda B. Viégas, Scott Golder, and Judith Donath. 2006. Visualizing Email Content: Portraying Relationships from Conversational Histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, New York, NY, USA, 979–988. DOI: http://dx.doi.org/10.1145/1124772.1124919

46. Lidan Wang and Douglas W. Oard. 2009. Context-based Message Expansion for Disentanglement of Interleaved Text Conversations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL '09)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 200–208. http://dl.acm.org/citation.cfm?id=1620754.1620783

47. Weixin Wang, Hui Wang, Guozhong Dai, and Hongan Wang. 2006. Visualization of Large Hierarchical Data by Circle Packing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. 517–520. DOI: http://dx.doi.org/10.1145/1124772.1124851

48. Franz Wanner, Thomas Ramm, and Daniel A. Keim. 2011. ForAVis: Explorative User Forum Analysis. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS '11)*. Article 14, 10 pages. DOI: http://dx.doi.org/10.1145/1988688.1988705

49. Merrill Warkentin and Peggy M. Beranek. 1999. Training to improve virtual team communication. *Information Systems Journal* 9, 4 (1999), 271–289. DOI: http://dx.doi.org/10.1046/j.1365-2575.1999.00065.x

50. Martin Wattenberg and David Millen. 2003. Conversation Thumbnails for Large-scale Discussions. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems (CHI EA '03)*. 742–743. DOI: http://dx.doi.org/10.1145/765891.765963

51. Furu Wei, Shixia Liu, Yangqiu Song, Shimei Pan, Michelle X. Zhou, Weihong Qian, Lei Shi, Li Tan, and Qiang Zhang. 2010. TIARA: A Visual Exploratory Text Analytic System. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*. 153–162. DOI: http://dx.doi.org/10.1145/1835804.1835827

52. J. J. Van Wijk and E. R. Van Selow. 1999. Cluster and calendar based visualization of time series data. In *Information Visualization, 1999. (Info Vis '99) Proceedings. 1999 IEEE Symposium on*. 4–9, 140. DOI: http://dx.doi.org/10.1109/INFVIS.1999.801851

53. Yingcai Wu, Shixia Liu, Kai Yan, Mengchen Liu, and Fangzhao Wu. 2014. OpinionFlow: Visual Analysis of Opinion Diffusion on Social Media. *Visualization and Computer Graphics, IEEE Transactions on* 20, 12 (2014), 1763–1772. DOI: http://dx.doi.org/10.1109/TVCG.2014.2346920

54. Panpan Xu, Yingcai Wu, Enxun Wei, Tai-Quan Peng, Shixia Liu, J.J.H. Zhu, and Huamin Qu. 2013. Visual Analysis of Topic Competition on Social Media. *Visualization and Computer Graphics, IEEE Transactions on* 19, 12 (2013), 2012–2021. DOI: http://dx.doi.org/10.1109/TVCG.2013.221

55. Ka-Ping Yee and Marti Hearst. 2005. Content-centered discussion mapping. *Online Deliberation* (2005).

56. J. Zhao, N. Cao, Z. Wen, Y. Song, Y. R. Lin, and C. Collins. 2014. #FluxFlow: Visual Analysis of Anomalous Information Spreading on Social Media. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (Dec 2014), 1773–1782. DOI: http://dx.doi.org/10.1109/TVCG.2014.2346922