Steering the Craft: UI Elements and Visualizations for Supporting Progressive Visual Analytics

Sriram Karthik Badam, Niklas Elmqvist, Jean-Daniel Fekete

Steering the Craft: UI Elements and Visualizations for Supporting Progressive Visual Analytics

Sriram Karthik Badam,[1] Niklas Elmqvist,[1] and Jean-Daniel Fekete[2]

[1]University of Maryland, College Park, MD, USA [2]INRIA, Saclay, France

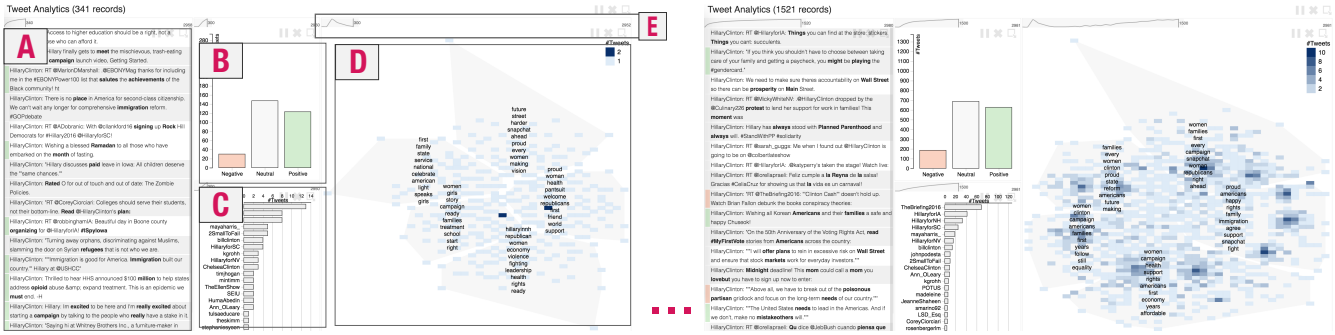

Figure 1: *The InsightsFeed tool for progressive visual analytics of Twitter (left): (A) a list of tweets, (B) a sentiment chart, (C) user popularity chart, (D) a map from a 2D projection of tweets with important keywords highlighted in each region, and (E) feedback and controls over the progression and computations. (Right) The interface is progressively updated when more data is processed.*

Abstract

Progressive visual analytics *(PVA) has emerged in recent years to manage the latency of data analysis systems. When analysis is performed progressively, rough estimates of the results are generated quickly and are then improved over time. Analysts can therefore monitor the progression of the results,* steer *the analysis algorithms, and make early decisions if the estimates provide a convincing picture. In this article, we describe interface design guidelines for helping users understand progressively updating results and make early decisions based on progressive estimates. To illustrate our ideas, we present a prototype PVA tool called* INSIGHTSFEED *for exploring Twitter data at scale. As validation, we investigate the tradeoffs of our tool when exploring a Twitter dataset in a user study. We report the usage patterns in making early decisions using the user interface, guiding computational methods, and exploring different subsets of the dataset, compared to sequential analysis without progression.*

Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Information Interfaces]: User Interfaces—GUI

1. Introduction

While data analysis has scaled dramatically in the last decade, this scalability has only impacted "confirmatory" data analysis, or model-based analysis, i.e. analyses where the structure of the data is known in advance, as well as the best algorithms for this analysis. For exploratory data analysis [Tuk77] and its modern incarnation *visual analytics* [Tho05, KAF*08], scalability remains limited due to the latency of traditional analysis systems. These systems typically yield long response times when analyzing large datasets or when using complex algorithms. Such latency is problematic in visual exploration since humans cannot maintain their attention when system response time exceeds approx. 10 seconds [Mil68, Shn84] and even small delays can affect their insights [LH14].

To overcome the latency issue, data analysis systems that deliver improving estimates of the results of computations have been introduced [KCL*17, MPG*14, PLvdM*16]. These systems engage the analyst during long computational processes by progressively visualizing their intermediate results and supporting interactive exploration by filtering data, as well as changing the parameters of the computation. This approach has been called *progressive visual analytics* [SPG14] (PVA) and shown to be more effective than traditional sequential (non-progressive) analysis systems for realistic tasks [ZGC*16]. Important reasons for the effectiveness of PVA include (1) reduced latency, (2) better transparency of how the computational methods work [MPG*14], and (3) support for early decision-making, either for making a final decision or for ter-

minating misguided analyses early. In addition, PVA ensures these advantages irrespective of the scale of the data and computation.

In this article, we focus on the human side of the PVA paradigm, especially in understanding (1) how visualizations can be adapted to better support PVA, and (2) how the user interface of PVA systems should be designed to provide the feedback and control needed to make early decisions reliably, and its effects on sense-making. We answer these research questions through an analysis of the design of a PVA tool called INSIGHTSFEED that we developed for analyzing large Twitter datasets. InsightsFeed adapts the traditional visual analytics pipeline [KKEM10] to incorporate intermediate results and feedback about progress from long computations, while providing easy controls to change the parameters of the progressive computations as well as supporting interactive filtering of visualized data. The InsightsFeed interface helps the data analyst understand the tweets, sentiments, as well as keywords discussed within tweets that are visualized using multidimensional projection algorithms to create a semantic map. Overall, we make three major contributions towards developing PVA interfaces by presenting,

1. UI and interaction design guidelines for new PVA systems;
2. An exploration of the design space through InsightsFeed for augmenting visualizations such as line charts and scatterplots to provide feedback and control needed for PVA;
3. Observations from a user study evaluating how the UI elements and visualizations capturing progression in our PVA interface are used to develop insights in contrast to an instantaneous interface.

2. Motivating Example

To understand the potential advantages of PVA, let us consider a data journalist who wants to understand the conversations on Twitter about the presidential candidate debate in the U.S., and write an article about it. For this analysis, tweets are collected during the debate using the Twitter API. A traditional analysis involves classifying sentiment, identifying popular users, and finding important keywords that stand out to make sense of the conversations [TSSW10, YGRP12]. This analysis process takes a time proportional to the number of tweets; and depending on the complexity of the computations, it can take hours. In contrast, our InsightsFeed system uses a progressive approach where the dataset is loaded by chunks of hundreds of tweets and the intermediate results of the analyses are updated every few seconds on the screen.

InsightsFeed presents four different views (Figure 1): (A) A list view of tweets with keywords highlighted, (B) sentiment histogram that captures the number of tweets with positive, neutral, and negative sentiments, (C) popular users bar chart that captures users who tweet the most, and (D) a tweet map created by projecting tweets on a 2D surface and placing similar tweets close-by [PPM04]. The computational backend of InsightsFeed works incrementally with chunks of tweets and the frontend updates the visualizations while providing (E) *feedback* of the progress and quality of the estimates, and *controls* to steer the computational methods.

Our data journalist working with InsightsFeed can get started with the analysis as soon as the interface starts. Within a few iterations of the progression, she immediately sees who the top users are and observes that most of the tweets have a negative sentiment. She also sees that "spending", "Iraq", and "war" are among the important words frequently appearing in the tweets and appear in separate regions in the tweet map (Figure 2). However, at the same time she also sees that the quality of the computational methods—sentiment analysis, popularity analysis, map generation—is changing, using the quality line chart on top of the visualizations. This line chart augmented with an interactive progress bar provides feedback of the progress and quality, and history navigation control. She then moves the playhead on the progress bar of 'popular users' backward to get a quick animation of the previous states and determine how the top three popular users evolved. She keeps the progression going and observes the sentiments and keywords in the tweet map. Halfway through the progression, she identifies that the labels in the map are similar across regions, so she steers the computation by increasing the number of regions (clusters) to generate differentiated labels. Steering controls can be accessed by clicking on the gray button on the top-right of visualization panels. Next, she finds that words "tax", "economy", "cuts" occur frequently with "spending" in the map, as well as "veterans", "care", and "needs" with "Iraq" and "war". She sees that the quality of the tweet map has now stabilized and decides to observe the keyword sentiments. She selects the corresponding regions on the map to update the visualizations with these tweets. She finds that tweets from top users with these specific keywords have a negative sentiment capturing their stance against the policy decisions related to these words.

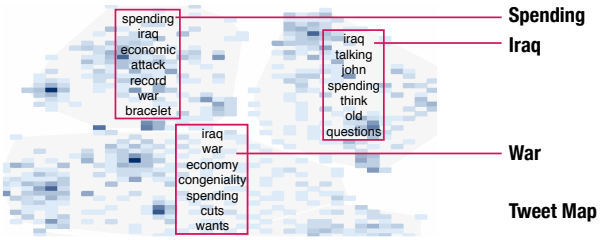

Figure 2: *Early stage in the progression of the tweet map. Three important keywords are found by the data journalist.*

After seeing a stable progression following her early observations, she decides to check back near the end of the progression. She starts writing her article and quotes the representative tweets from the top users. Near the end, she finds a few more specific words associated with the regions on the map she was interested in. She adds more details to her article about "defense", "policy" and "Bush" to create a story on the public's reaction to the Bush administration's policies and spending in relation to the Iraq War.

In this example, the analyst has been able to start her exploration without delay; she tracked the progression until she could make decisions confidently using the quality and progression feedbacks.

3. Background

We introduce here the concept of progressive visual analytics and its HCI-related aspects: evaluations and requirements.

3.1. Progressive Visual Analytics

Hellerstein et al. [HHW97] introduced the concept of progressive data analysis; he showed that, when computing aggregate queries in a database (e.g., mean of a column), the database could return estimates that improve over time instead of waiting until the computation finishes. The concept was meant to avoid latency and monitor the results through simple visualizations with confidence intervals, allowing users make confident judgments for early decisions.

The principle of reducing the latency by computing improving estimates has then become popular under multiple names (iterative, online, streaming, any-time) [FP16]; the term "progressive" avoids ambiguity. Furthermore, early progressive systems have provided interactions for *steering* the computations [MvWvL99]. Steering involves interactively changing parameters of an algorithm while it runs; it encompasses multiple types of operations, from filtering data if the analyst wants to focus on specific values, to tuning algorithm parameters and behavior dynamically. Therefore, progressive data analysis provides both feedback and control: feedback mainly through a sequence of improving estimates and control mainly through steering parameters. Many recent research articles have introduced adaptations of algorithms for progressive feedback and steering (e.g. [BP07, PLvdM*16, SPG14, WM04]).

Stolper et al. [SPG14] introduced the term *progressive visual analytics* as the integration of progressive data analysis and visual analytics [KAF*08]. This allows analysts to explore large amounts of data by visualizing partial results, monitoring improvements as they arrive, and performing new rounds of exploratory analyses without waiting for analyses to complete. Their prototype system includes a progressive version of the SPAM algorithm [AFGY02] that computes and visualizes correlations in patient records.

3.2. Evaluations and Requirements

Progressive visual analytics has implications for perception, cognition, and more generally human factors. Its effectiveness has been shown by multiple studies. Fisher et al. [FPDS12] showed that "Online Aggregation," the progressive computation and visualization of statistical measures such as mean and variance, was effective: participants of a controlled experiment were able to make early decisions reliably based on progressively computed estimates, as well as error bars and confidence intervals. Glueck et al. [GKW14] have shown that progressively loading large time series for exploratory visualization allowed early exploration and quick and accurate report of important insights. Zgraggen et al. [ZGC*16] studied the effect of progressive visualization compared to an ideal *instantaneous* condition with no perceived delay, and a *blocking* condition that updates after 6 seconds or 12 seconds. They showed that users performed equally well with instantaneous and progressive visualizations in key metrics such as insight discovery rates and dataset coverage, while blocking visualizations had detrimental effects.

Designing PVA interfaces implies following important requirements to overcome the algorithmic issues and optimizing the human aspects. These requirements have been elicited by Stolper et al. [SPG14] and Hetzler et al. [HCPT05], and more generally by Mühlbacher et al. [MPG*14]. We summarize them in Table 1 following the 2D organization from Mühlbacher et al. [MPG*14].

	Result	Execution
Feedback	R1: Meaningful partial results R2: Structure-preserving intermediate results R3: Retaining cognitive workflow on updates R4: Minimized distraction during updates R5: Cues for new results R6: Aggregated information R7: Uncertainty R8: Provenance	R9: Aliveness R10: Absolute progress R11: Relative progress
Control	R12: Full interactivity R13: On-demand refresh to update when ready R14: Steer results	R15: Cancellation R16: Prioritization

Table 1: *Requirements from the related work [SPG14, HCPT05, MPG*14] for creating PVA systems in terms of the user feedback and control provided for the results and executed processes.*

While these requirements are developed to be generally applicable to PVA scenarios, the extent of their application varies with the context. For instance, when visualizing simple statistical measures [FPDS12], the need for steering results, providing controls, and provenance is minimal compared to a PVA interface running multiple flexible algorithms (e.g., similarity search [SASS15]).

3.3. PVA Systems

A key challenge for any PVA system is instantiating these requirements into concrete interactions, UI elements, and visualizations. Previous PVA systems have been designed with part of these requirements in mind. For instance, (1) R1 and R3 come from Stolper et al. [SPG14], (2) R4, R5, R12 and R13 are taken from Hetzler et al. [HCPT05] (adapted to PVA to an extent by Stolper et al. [SPG14]), and (3) the rest are from Mühlbacher et al. [MPG*14]. However, they share common aspects in supporting some of these requirements and contrast in terms of others (Table 2). We therefore compare four PVA systems based on Table 1: SampleAction [FPDS12], Incremental Visualization [SASS15], ProgressiveInsights [SPG14], and VizDom [ZGC*16].

Feedback for computation results: All of the systems comply with R1 by showing partial results of progressive computations. Cues of new results (R5) are shown in ProgressiveInsights by coloring new points corresponding to new results in orange and showing lines between list items where new items will appear. All the interfaces comply with R6: the visualizations show aggregated values because they are designed for scalability. Some capture uncertainty (R7): SampleAction [FPDS12] adds green dots to each bar in its chart to convey error bounds; VizDom [ZGC*16] shows error bars that shrink as the uncertainty decreases during the progression. Incremental Visualization [SASS15] provides provenance information (R8) using a data-flow diagram of the computation.

Feedback for execution: SampleAction [FPDS12], Incremental Visualization [SASS15], and VizDom [ZGC*16] comply with R11 by showing relative progress and quality over time through, respectively, a percentage value, a small circular progress bar, and a line

	Update Cues (R5)	Uncertainty (R7)	Provenance (R8)	Relative Progress (R11)	Interactivity (R12)	Refresh (R13)	Steering (R14)	Controls (R15/R16)
SampleAction [FPDS12]		✓		✓				✓
Incremental Visualization [SASS15]			✓	✓			✓	✓
ProgressiveInsights [SPG14]	✓				✓	✓		
VizDom [ZGC*16]		✓		✓	✓			

Table 2: *Differences across existing PVA systems in terms of their support for different features that enable and take advantage of the PVA paradigm. Note that some PVA interfaces compensate for missing requirements through support for others, as discussed in Section 3.3.*

chart. Note that progress information (R10, R11) together with the evolution of uncertainty (R7) allows for assessing the trustworthiness of the visualized results for decision making. PVA systems therefore balance their UI features to support more than one of these requirements at the same time. For example, quality as the relative progress measure in Incremental Visualization [SASS15] also implicitly captures the uncertainty the result.

Result and execution control: Hetzler et al. [HCPT05] discuss about "providing full set of interactions that are available with regular or non-dynamic datasets" (R12). In PVA, filtering options to focus on subspaces are supported by ProgressiveInsights [SPG14] and VizDom [ZGC*16]. R13 "on-demand refresh" is a feature of ProgressiveInsights but other systems refresh the views along with the computation. We return to this specific point in Section 7.3. Incremental Visualization [SASS15] provides a set of control buttons that can change the parameters to steer the algorithms (R14) on the fly or stop the progression entirely (R15), while SampleAction [FPDS12] allows pausing and stopping the processes (R15). ProgressiveInsights [SPG14] partially supports result steering (R14) by starting completely new analyses with different parameters and reverting to the previous in a "run stack" if needed.

Overall, the Incremental Visualization interface complies with a lot of the requirements to take advantage of the PVA paradigm. However this comes at a cost as it devotes only a small portion of its screen real-estate to show the actual computation results. Therefore a main challenge of PVA is *complying with the requirements* and *providing a consistent set of controls and feedback* while *managing the screen real-estate based on the analyst's degree of interest*.

4. The InsightsFeed System

We created the InsightsFeed system to explore large textual datasets from social media such as Twitter, Reddit, and Facebook using the PVA paradigm. While the system is generally applicable to other social media, here we focus on Twitter. The datasets typically consist of tweets (short texts in general), replies, retweets, author information, and some metadata (e.g. timestamp, location, tags).

As explained in our motivating example, the InsightsFeed system consists of four connected views (Figure 1). These views are driven by an integration of progressive computations and progressive visualizations, which together make up a PVA model. Here we explain the progressive visualizations and their computational methods, and then describe the user interface elements, visual representations, and controls added to support the PVA paradigm, following the requirements listed in the Background section.

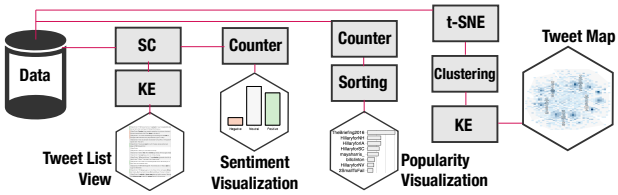

Figure 3: *Dependency graph of the InsightsFeed views. KE is keyword extraction and SC is sentiment classification.*

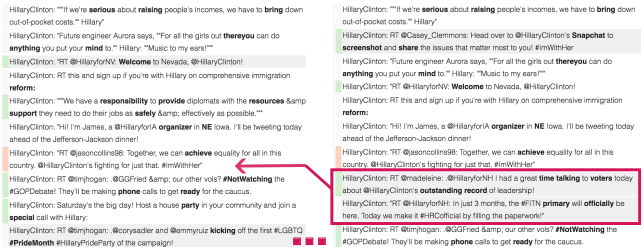

Figure 4: *Two instances of the tweet list view, during progression, with keywords highlighted. New tweets are inserted based on the timestamp order and highlighted with a gray background.*

The computational methods used in InsightsFeed are adapted to have a progressive nature through simple mechanisms to work incrementally with chunks of data, build on data structures from previous steps, and maintain stability. However, some of the algorithms used—t-SNE and k-means—need further modifications to their process logic to be fully progressive. While progressive algorithms would improve the computation throughput, the UI aspects explored in this paper would remain the same since our visualizations apply aggregation and sampling scalable to large datasets.

The system starts by reading a dataset, tweet by tweet, from a local file. The tweets are stored in random order to improve the convergence of progressive algorithms. Each tweet is pushed into a chunk; when it is of a particular size, it is passed to the computational methods that send the results to the visualizations (Figure 3).

4.1. Tweet List View

This view shows the tweets in a list sorted by tweet timestamp (Figure 4). The important keywords in the tweets are highlighted and the tweet sentiment is shown using color: *red* for negative, *gray* for neutral, and *green* for positive. The computational methods driving this view are sentiment classification and keyword extraction.

Sentiment Classification: This is driven by VADER [HG14], a rule-based model for sentiment analysis. VADER is created with a sentiment lexicon that is sensitive to both polarity of words (positive or negative) and intensity of the sentiment, and is ideal for social media contexts. It scores a given sentence on a scale of $[-1, 1]$, corresponding to a range from extremely negative to extremely positive. The tweet is assigned a sentiment directly based on this score.

Keyword Extraction: The tweets are treated as documents; while it may appear odd to treat 140-character sentences as documents, twitter datasets following real-life events are often rich with keywords related to the event. Therefore, inverse document frequencies can be computed when enough chunks have been loaded. The term frequencies (tf) and inverse document frequencies (idf) of the words are computed to generate a tf-idf score for each word in the tweets [SJ72]. Stopwords are removed, and only nouns, adjectives, verbs, and adverbs in their base form are considered. The words are then sorted by their tf-idf scores and the top words are picked as keywords. The chunk containing the tweets, authors, timestamps, keywords, and sentiment information is passed to the list view.

Progressively visualizing tweets: To minimize disrupting the user's workflow (R3, R4, R5), the tweet list is updated with each chunk by a two-step process: (1) each tweet is inserted in timestamp order and highlighted with a gray background, and then (2) tweets from previous chunks are deleted starting with the oldest chunk until all the new tweets fit into the view. Keywords are highlighted and sentiment is shown as a colored band beside each tweet.

4.2. Sentiment Visualization

The sentiment visualization on our PVA interface is a bar chart showing number of positive, negative, and neutral tweets.

Counter: The computational method driving this visualization is a counting module that keeps track of the frequency of each sentiment. To do so, it utilizes the sentiment classifier used for the Tweet list view to get the sentiments associated with the tweets in a chunk. The sentiment counts are then passed to sentiment visualization.

Progressively visualizing sentiment frequency: The sentiment bar chart is updated with each chunk through staged animation [HR07] (R3, R5): the axes are updated first, followed by the individual bars. To avoid continuous axes updates (R4), they are updated to twice their range when a bar reaches the top of the chart.

4.3. Popularity Visualization

The popularity visualization is a horizontal bar chart showing a sorted list of Twitter users based on their activity—number of tweets and number of mentions in replies or retweets.

Sorting: For each chunk, user names are collected from the tweet content and a counter similar to the sentiment counter is used to compute the user frequency. This list of user frequencies is sorted in descending order using a stabilization method: the previous order is used as initial state for the stable sorting algorithm to reduce the number of changes, since users will eventually keep their rank.

Progressively visualizing popular users: Similar to the sentiment bar, popularity visualization follows a staged animation (R3, R5): (1) the axes are updated, (2) the user locations are changed based on frequencies and new users are added, and (3) the size of the bars are updated. Only users who fit in the chart height are shown. Frequent axis updates are avoided similar to the sentiment visualization (R4).

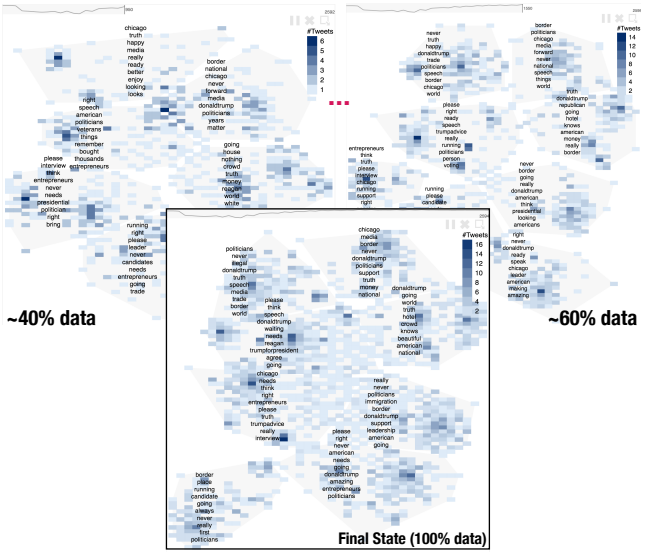

Figure 5: *Three instances of the tweet map. Tweets are projected, aggregated into heatmap, clustered, and labelled. The labels are generated by extracting the top keywords in each cluster region.*

4.4. Tweet Map

Our main progressive visualization, covering most of the interface, is a Tweet map (Figure 5): a 2D projection of all the tweets where similar tweets are positioned close-by, according to a semantic similarity measure. Over the Tweet map, areas called **landmarks** highlight the clustered regions of the 2D space. The map is driven by two computational methods: t-SNE projection algorithm [MH08] to create the map, and clustering to create the landmarks.

t-SNE Projection: The tweets within each chunk are projected using a similarity measure based on the words within each tweet. Stopwords, hashtags, user names, URLs, punctuations, and emoticons are stripped from each tweet in the chunk to extract the core text. From this text, the nouns, adjectives, verbs, and adverbs are extracted by parts of speech tagging and converted into their base form by stemming to apply the similarity measure. Following this, similarity between two tweets is determined by counting the number of matching words between the tweets; two words are matched when they are either the same or have common synonyms. We use the t-SNE manifold embedding algorithm on the dissimilarity matrix (inverse of similarity) created for all the tweets to compute their 2D embedding (positions). The positions are initialized before running t-SNE based on the positions of the tweets from previous chunks and the similarity metric. We initialize the position as the center of the nearest neighbors, ensuring that positions of the projected tweets do not change drastically with each iteration (R4).

Clustering: The locations of the tweets from the projection method

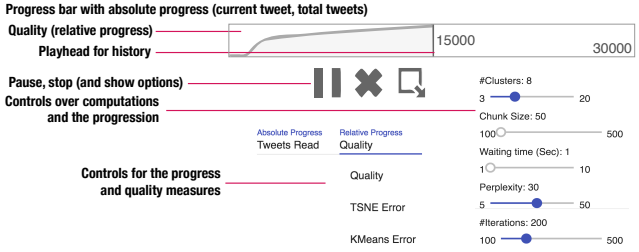

Figure 6: *Feedback and controls added to each visualization to track the progression and control the computations.*

are passed through a k-means clustering algorithm [KMN*02] (with k=5 as the default) to find the clusters of points referring to similar words; these clusters define landmarks. The tweets in each cluster are passed through the keyword extraction procedure, same as the one used in the tweet list, to find the important keywords within each cluster. At the same time, to avoid overplotting, the projected points are accumulated into a 2D histogram to create a heatmap, an aggregated visualization that scales. The heatmap with the tweet projections, the clusters, and the landmark label information are passed to the Tweet Map view on the interface.

Progressively visualizing the tweet map: Given that this view captures multiple information types—tweet projections, clusters, and landmark labels—the progressive visualization is updated in multiple steps (R3, R5): (1) the heatmap smoothly transitions in cells that changed, (2) cluster is changed next, and, (3) finally, the labels are updated. Users are informed of the update with a small blinking rectangle in the progress bar on top of the view while minimizing distraction (discussed in Section 4.5).

4.5. UI Elements for Feedback and Control

Based on the guidelines from Mühlbacher et al. [MPG*14], all the progressive visualizations are coupled with additional UI elements and small visualizations to provide feedback and control (Figure 6) over the computational methods.

Progress Bars: Each progressive visualization has a progress bar that captures the number of tweets processed to get to the current state of the progression (R10). However, recall that the entire system is driven by a dataset reader that reads tweets line-by-line. Therefore, the total number of tweets in the dataset used for populating this progress bar is also a progressive measure that is estimated based on the total lines read, total bytes read, and size of the file in bytes. Each visualization in the system can have a different progress as the computations take different amounts of time and the user can influence the processing speed. A blinking rectangle is placed beside the progress bar to highlight the current chunk being processed by the computations. This satisfies requirements R9-R10, as the user is informed of the progress and the aliveness. From an UI design point-of-view, other absolute progress measures can also be used for this feedback, based on (1) time spent and total time left, (2) bytes read and total size of the dataset, and even (3) the error metric of the computations and target error.

Quality (Stability) Visualizations: The user needs an indication of the quality or uncertainty (R7) of the current state of progression: the relative progress (R11). For consistency across the progressive visualizations within the UI, we decided to use a quality/stability metric, which has specific properties: (1) the quality metric will always increase when the computation is improving, (2) it will decrease when the quality declines, and (3) it will otherwise be stable at a particular value. Based on this, we created the quality metrics from our computational methods and visualized them as a small line chart on top of each visualization:

Tweet map: inverse of the projection error obtained from the projection algorithm.
Tweet list: number of new keywords per chunk.
Sentiment: cumulative change in the sentiment distributions.
Popularity: number of new users added per chunk.

The metrics for the tweet list, sentiment visualization, and popularity visualizations are cumulative as there is no meaning to performance declining in their case.

Progression and Computation Controls: To control the progression, each visualization provides:

- Play, pause, and stop buttons to control the progression and interactively explore the visualized data without worrying about changes (R13, R15).
- A playhead on the progress bar that can be moved backwards to see an animation of the previous states (R5).
- Sliders that can set the parameters of the computational methods including the chunk size for each visualization, update speed, and internal parameters of projection and clustering (e.g. number of clusters). Changes to these parameters are reflected in the next iteration, thus steering the computations (R14, R16).
- Options to change the absolute progress and quality.

4.6. Interactions

Similar to other VA systems, InsightsFeed provides interactions to explore the data visualized on the interface. The interactions are centered around the tweet map (Figure 7): (1) the keyword distribution within each cell in the map is shown on *hover* and (2) the visualizations are updated when a region is *selected* on the map.

When a region is selected by a user, the entire interface is paused to avoid updates and support exploration of the data currently shown. It reverts back to progressive mode when the user clicks the "play" button on the interface (R13). Therefore, we adapt R13 slightly by providing two modes: a *monitoring* mode where users see the progression and an *interaction* mode where the progression occurs in the background and the interface is refreshed on demand.

4.7. Mapping Requirements to User Interface

InsightsFeed satisfies the requirements presented in Section 3.2. It provides meaningful partial results that are structure-preserving for all the progressive visualizations (R1, R2). This is done by shuffling the dataset ahead of time. The updates are made through a transition process for each visualization depending on its content (R3, R4, R5). Our visualizations (except Tweet List) use aggregation (e.g., heatmap) (R6) and also extract important information (e.g.,

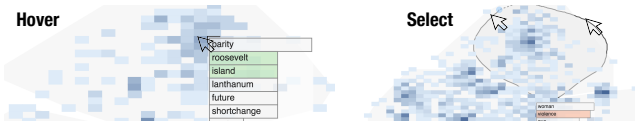

Figure 7: *Interactions with the tweet map: hover on a cell to get the keywords, and select a region to update the interface.*

top users) to provide representations **scalable** to large datasets. Progress bars are attached to each view along with an aliveness indicator and a quality curve to convey execution feedback (R9, R10, R11) and uncertainty (R7), reinforced by history navigation using the playhead. The user interactions (Figure 7) allow exploration of subspaces (R12) and further support visual analysis. The control options are designed to update results when wanted (R13), steer computations (R14), and cancel/update the execution (R15, R16).

4.8. Implementation

The InsightsFeed implementation used web technologies—HTML, CSS, and JS—for the interface, D3 framework [BOH11] for the visualizations, and a Python Tornado server for executing computational models. The computational methods are implemented using Python scikit-learn [PVG*11] and NLTK libraries [BKL09].

5. User Study

Recent work [ZGC*16] showed that progressive visualizations lead to a similar number of insights as instantaneous visualizations, while blocking delays the sensemaking process [LH14]. However, the effects of progression, the associated uncertainty, and feedback and control options on the analyst's sensemaking process are unclear. We therefore conducted a user study to compare sensemaking on our PVA tool with a traditional, instantaneous VA tool, where all results are instantly available. We focused on observing how analysts build their confidence when working with approximate answers using the progressively updating UI elements and visualizations in our system. While our t-SNE and k-means implementations are not fully progressive, they still provide partial and incremental results by chunking data and using scalable aggregated representations. Since our focus is on the process of sensemaking (not just the outcome), the current InsightsFeed implementation applied to a practical Twitter dataset can reveal ecologically valid outcomes.

5.1. Participants

We recruited 10 participants (2 female, 8 male), who had previous experience with data visualizations. Participants were between 18 and 45 years of age and recruited from our neighboring HCI/visualization research labs. All participants self-reported as proficient computer users. Data analysis experience varied across participants: all were experienced with creating charts for reporting, 3 participants worked in positions that require analyzing specific datasets, and 5 participants developed visualization tools.

5.2. Experimental Factors

We used interface type (I) as the single experimental factor:

PVA: The full version of our system, including progressive visualizations, widgets, and progress and computation controls.
IVA: A stripped-down version of our tool with no progression; the final state of the analysis is shown directly.

5.3. Dataset and Task

Given the global interest in the United States presidential election, we decided to pick datasets from the Twitter accounts of Hillary Clinton and Donald Trump. Since we were studying two different interfaces—progressive and instantaneous—we collected two sets of random tweets (about 2,900 each) from their accounts from April to December 2015. Having two datasets enabled a within-subject design. We created a list of five tasks (questions) for each dataset that involved each of the four views in the interface. All tasks were given at the start and could be answered in any order.

Q1 What is the most frequent sentiment in the dataset?
Q2 Who are the four users making the most tweets?
Q3 What are three most frequent words on the tweet map?
Q4 What is a representative tweet that captures the candidate's stance on immigration (Trump) or rights (Clinton)?
Q5 What are the words associated with "health" (Clinton) or "border" (Trump)? Pick three words and also identify the candidate's sentiment with them.

We consider these two datasets equivalent as they concern the same real-world event—the U.S. presidential election campaign—and the questions were related to topics typically discussed by the candidates. We ensured that the two sets of the tasks were approximately equally hard to answer. We also counterbalanced the combinations of dataset and interface across participants.

5.4. Procedure

During each experiment session, the participants started with their assigned interface and dataset. They then went through the training procedure for the assigned interface. For PVA, we explained the participants the four levels of features on the interface: (1) individual visualizations, hover and select interactions, and the animations during each iteration of the progression; (2) progress bars and quality visualizations on each visualization and what they represent in terms of the computational method behind the visualization; (3) playhead controls on the progress bars to go back in history and see previous states of the visualizations; and (4) controls over the computational methods and progression including processed chunk sizes, speed of progression, and number of clusters on the tweet map. The progression parameters are initialized based on the visualization content—Tweet Map updates less frequently than sentiment visualization to give the user more time to interpret it. Training on the PVA interface lasted about 15 minutes. Participants were then quizzed about features to make sure they understood them and the explanations were repeated. On the IVA interface, a similar procedure was followed to explain the visualizations. After finishing one interface and dataset, the procedure was repeated for the other.

After training, the participants began answering questions for the assigned condition. For the PVA interface, the progression of the visualizations was started immediately. Participants were encouraged

to answer the questions whenever they had an answer; they were asked how confident they were with their answer, if they would wait for more iterations to give a more confident answer, and what would improve their confidence. At the end of the progression, they provided their final answers. For the IVA interface, participants answered the questions immediately.

Following each task, the participants were asked to (1) write a short paragraph (3-5 sentences) about their findings, (2) provide their subjective feedback about each interface in terms of efficiency, ease of use, and enjoyability, and (3) express their experiences with individual features of the interface. Each session lasted up to 75 minutes. Participant interactions were recorded during the session and they were asked to think aloud when answering the questions.

5.5. Hypotheses

While the data collected is mostly qualitative, we were still interested in developing some hypotheses of the outcome. Our main hypothesis for the PVA condition is that the participants will generate a large number of early observations to answer the questions and develop them over the progression. This high-level hypothesis can be broken down into the following four underlying hypotheses:

H1: Progress bars with quality measure will be associated with confidence in answering questions.
H2: Progression history will help track any substantial changes.
H3: Control options, especially cluster size, will be used to guide the information shown.
H4: Final answers will be developed from early observations, and will be similar between PVA and IVA interfaces.

Due to the study's qualitative nature, we assume that an observed behavior *supports* a hypothesis if at least five participants follow it. Hypotheses are rejected if none use the corresponding features.

6. Results

Here, we discuss the usage patterns of our PVA system in the study.

6.1. Differences: Instantaneous versus Progressive

All participants correctly answered the questions, with correct reasoning, on the instantaneous VA interface. When using InsightsFeed, eight participants gave correct final answers to all questions. Two participants mentioned a few words that ended-up not being frequent because they gave their final answers about the frequent meaningful words at around 50% of the progression when the quality metric nearly stabilized, and chose not to revise their answers. *This behavior supports H4 as most participants answered correctly.*

There were also differences between the two interfaces in terms of the time taken to answer the questions. Participants were faster on the instantaneous interface (mean = 7:27 min, s.d. = 3:16 min) than the progressive (mean = 13:48 min, s.d. = 4:27 min) as expected. This includes time to read questions, interpret visualizations, and also talk their answers out aloud during the tasks.

6.2. Early Observations

All participants made early observations using the PVA interface, and most (except two) developed the answers during the progression. Each participant made at least two or more early observations for questions Q1, Q2, and Q3, and two or less observations for Q4 and Q5. Overall we found three forms of early observations:

Partial and confident answers: These are observations that partially answer the question. For example, the top user is @foxandfriends, and "politicians" is a frequent keyword. All participants except P4, P9, P10 gave partial answers for popular users at appx. 20% progression. They noted that the quality measure had not stabilized, but they observed that the top user was far ahead. Similarly, some participants (P1, P7) gave partial answers for frequent keywords (Q3) since some words were consistently appearing in all progression states even though their quality had not stabilized.

Complete but unconfident answers: These are observations that completely answer the question, but the participants were not confident about it. This pattern was followed by all participants when answering the popular sentiment (Q1); they would answer this at 10%-20% progression but mention that they are not confident. Participants P9 and P10 gave early answers with low confidence for popular users (Q2) by observing the quality metric and previous states when some users moved to the top. Some participants (P3, P5, P6, P8) gave this type of answers to Q3, Q4, Q5 around 40%-50% progression when the tweet map was stable but low quality.

Complete and confident answers: These were answers where participants were very confident. They were given after 40% progression for sentiment (Q1) and popularity (Q2), and after 60% progression for remaining questions by all participants when stability was reached in the quality visualization and the history of previous states (explored by P1, P3, P5, P7, P8) suggested that the answer was consistent across the progression.

Only one participant (P1) gave a partial and unconfident answer at 5% progression. Questions Q4 and Q5, which were answered by selecting regions in the map with the appropriate keywords, had the least early observations. Participants would rather wait for stability before answering them. Beyond these types, some participants (P1, P5, P9) made **serendipitous observations** on the progressive interface. For example, while searching for tweets about "rights", P9 found different contexts ("gun", "women", "equal") appearing at different steps of the progression in the map and further explored these regions to find more representative tweets.

6.3. Estimating Confidence

All participants associated some confidence with each early observation. This was mainly done in two ways:

Following the stability: Most participants (P2, P4, P5, P6, P9, P10) used a quality-centric assessment of their confidence. They would directly quote the quality and its stability when making early observations. Thus, these participants waited for quality to reach a near-stable state before starting to answer. In some situations, they would only look at previous progression states when there was a sudden change in quality to form a soft **confidence interval** around the visualizations and provide early answers. *H1 is thus supported.*

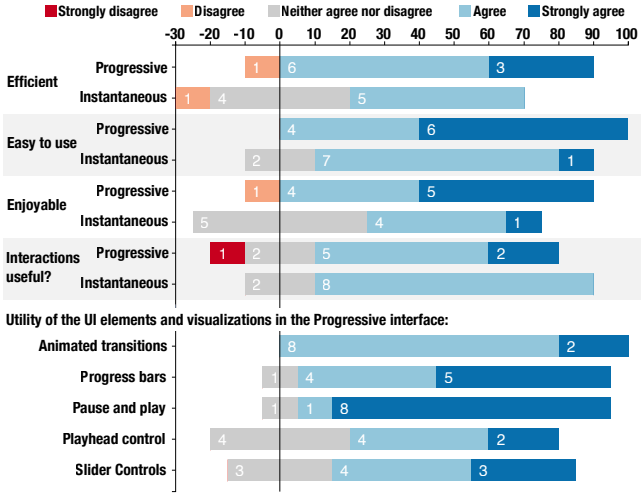

Figure 8: *Participant response counts for the Likert scales concerning the usability of the progressive (PVA) and instantaneous (IVA) interfaces, as well as the utility of the features.*

Tracking changes: Some participants (P1, P3, P7, P8) tracked the changes during early stages of progression when quality measure is not yet stabilized to answer the questions better. Even after stability, these participants would move the playhead to check the history before answering the questions. They accordingly needed more time to derive early answers, but these participants were in general more confident with their answers since they had a better understanding of the progression. This was also dependent on the visualization type: for the bar charts, participants could easily observe the change in the heights when moving the playhead, whereas for the tweet map and list view, they took longer to see what had changed in the heatmap. P9 mentioned that she wanted to see some common keywords on the map even if they were no longer important to help her keep track of what the region previously corresponded to. *H2 is thus partially supported since four participants tracked changes by navigating through the progression history using the playhead.*

6.4. Steering the Progressive Visualizations

Seven participants (P2, P3, P4, P5, P6, P7, P9) manipulated the computation controls, especially the number of clusters. Only P3 changed the chunk size to see less updates on the tweet map. The number of clusters was changed for two different reasons, both of which were interestingly not related to correcting some unimportant results [MPG*14] but rather to change the amount of detail.

Controlling the landmarks on the map: Four participants (P2, P3, P5, P7) decreased the number of clusters to answer questions regarding the frequent keywords on the map during early stages. They mentioned that having a large number of clusters when the map only contained few hundreds of tweets led to many unimportant keywords appearing on the visualization. They would then read the popular keywords from the updated landmarks to answer Q3.

Exploring more details and associations: To answer the question about the keywords associated with particular words such as "health" and "border," participants had the option to select the regions of the map corresponding to the words to see the selected tweets and their sentiment. However, some participants (P4, P5, P6) would increase the number of clusters (to 10 or more) to get more specific clusters and find the associated words without having to select the tweets in the map. This specific style of clustering captures how advanced users of PVA interface can take advantage of the control over the computational methods to change the amount of information based on their preference. *H3 is thus supported since multiple participants took advantage of the control options.*

6.5. Subjective Ratings

After each task, the participants rated the interface on three metrics—efficiency, ease of use, and enjoyability—on a Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree). These questions were presented as, "Is this interface efficient for answering the questions?" The utility of different UI features is presented on a similar scale. Overall, both progressive and instantaneous interfaces (Figure 8) received similar ratings on the efficiency, ease of use, and enjoyability scales. Almost all participants agreed to these questions. We believe that the minor differences are due to the presentation order of the interfaces; when presented with progressive first they found this new interface to be more interesting as they come up with more (early) answers. The interactions (hover and select), while similar across interfaces, received slightly different ratings from a few participants. P5 who strongly disagreed with the utility of the interactions on PVA mentioned, "*I used the controls [changing clusters and playhead] to answer the questions.*"

The features in InsightsFeed were used to different extents (hence differences in Figure 8). All participants found the animations useful. When asked if the animations were hard to discern, the responses were mixed. Participant P5 said, "*Because the animation is slow, we could read every word on each cluster*", and P7 said, "*When it changes frequently it can be a bit hard, but being able to pause helps a lot.*" For the rest of the features, participants who did not use them often gave a neutral rating. P9, who did not use the playhead control to see the history, said that "*I would use it only to 'verify' how the algorithm works. [But] I didn't use it because I am interested in an overview of all iterations at the current state.*"

7. Discussion

In this section, we explain our results, discuss their limitations, and revisit the guidelines for PVA systems.

7.1. Explaining the Results

All participants made multiple early observations before reaching a final answer. However, the behavior diverged on the *type* of early observations. The type of observations and confidence mainly relied on the feedback elements in InsightsFeed—the visualized quality and the progression stability. The early observations may also depend on the individual preference—some people are more comfortable giving partial answers than uncertain ones. While early observations were eventually developed into final answers, complete but uncertain answers sometimes revealed additional information

about the dataset. For example, early answers about frequent words sometimes indicated words that were frequent in only part of our dataset, e.g. "golf" and "business" for the Trump dataset. Characterizing which types of early observations are better in the long run is an important future step for understanding the PVA paradigm.

An interesting result was that seven out of ten participants used the cluster controls in our PVA interface to tune the landmarks on the tweet map. Surprisingly, some did this to avoid interacting with the tweet map and just see more information on the landmarks. However, this can also have an unintended effect of overlaying more complex semantic structure on the map, since distinguishing map regions is based on assigning a meaning to keywords generated by the automated algorithm. Overall, this feature was useful to configure the visualization complexity to a comfortable value.

7.2. Limitations

- The datasets in our study were small. However, we believe that they were complex enough to create realistic tasks and observe the usage of our PVA interface. Furthermore, the visualizations and UI elements in InsightsFeed use aggregation and sampling to scale to larger datasets. Having said that, there may be differences in the time spent on monitoring progression and interactive exploration in case of larger datasets. However, our study results about the sensemaking process in PVA, including the evolution of the early observations, estimation of confidence, and the usage of UI elements, would still remain meaningful in these settings.
- Our experiment focused on a particular set of tasks rather than on open-ended exploration. While we believe that these tasks were representative of a typical analytical process, future studies should focus on high-level exploration.
- The experiment used cached results to avoid differences across participants, so projection parameters were not changed.
- Similar to Zgraggen et al. [ZGC*16], our study included no anomaly detection, which can be challenging on PVA interfaces.

7.3. Revisiting the PVA Requirements

Building on our qualitative results, we revise some of the PVA requirements to be more concrete and present two new requirements:

R8-revisited: Flexible PVA systems need to show provenance information on demand, in particular the analysis pipeline. Static applications can describe the pipeline offline.
Reason: The InsightsFeed pipeline is described in Figure 3; showing that diagram explains part of the provenance. More flexible systems, where the analysis pipeline can be configured dynamically, should provide visualizations of the pipelines [FP16, SASS15].

R13-revisited: Support two PVA modes: a monitoring mode, where progression spawns constant animated updates, and an exploration mode, where updates are added on-demand.
Reason: This helps analysts avoid dealing with progression when trying to interpret the current visualized data. Participants from our experiment found the pause and play options to be very helpful for this reason. However, the exploration mode with steering controls can support mixed-initiative interactions [EFN12, KLTH12]. Therefore guidelines from that field [Hor99, Hor07] should be considered when designing PVA interfaces with dual modes.

New R17: Provide similarity "anchors" in complex visualizations to help the user maintain his mental map.
Reason: Our tweet map has multiple layers of information: heatmap densities, cluster regions, and landmark labels; tracking changes on all layers is challenging. As P9 said, an indication of what keywords were previously there by permanently adding them to the interface can be helpful. This extends to creating **visual anchors** on visualizations that help users keep track of the progression. However, these visual anchors can be based on the primary information (the landmarks) or extend to multiple layers (landmarks+regions) on the visualization. Designing these anchors is an important research topic that needs to be closely studied.

New R18: Use consistently visualized quality measures for the visualizations to simplify comprehension.
Reason: Participants often cited their confidence for early observations based on the quality measures and they also found the progress bars and quality measures to be useful. However, note that we relied on global quality measures represented consistently across progressive visualizations. Alternatively, local quality measures that define, say, the quality of the t-SNE embedding in different 2D regions, can also guide the user's observations about the data underlying those regions. Exploring more quality metrics with consistent representations remains to be a part of our future work.

7.4. Implications

InsightsFeed targets Twitter data, but we consider it a generic PVA system and believe that our study shows how the features within PVA systems play a role in guiding the user in developing early observations, confidence, and steering computations. We observed specific types of early observations guided by the quality and stability feedback within our InsightsFeed interface. We also noticed how providing direct access to parameter steering through control options (Figure 6) changed the user strategies (Section 6.4).

The list of requirements presented in this paper is rather large (18). However, we presented a system that implements them all by *providing a consistent set of controls and feedback* while *managing the screen real-estate*. Our system together with our findings conveys the unique affordances of having different features in a PVA interface and explains how new PVA interfaces should be designed.

8. Conclusion and Future Work

We have presented an operationalization of requirements for progressive visual analytics using a visual text analysis tool called InsightsFeed. The purpose of our design is to streamline and standardize the plethora of such requirements found in the literature and to determine best practices for user interface and visualization design for PVA. Our contributions center on visualization adaptations and on a *progression toolbar* that provides both a visual representation of the progress and convergence of the underlying computation, as well as controls for pausing, refreshing, and navigating the computational history. We also presented results from a user study comparing analytical performance using InsightsFeed versus a traditional non-progressive tool.

Our study helps understanding the design requirements for PVA as well as giving strong indications on future research directions,

including interaction models for steering computation, visual representations for confidence and convergence, and novel aggregation methods for scalable visualizations.

Acknowledgements

We thank the anonymous reviewers for their feedback. This work is supported by U.S. National Science Foundation award IIS-1253863. Any opinions, findings, and conclusions or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the funding agency.

References

[AFGY02] AYRES J., FLANNICK J., GEHRKE J., YIU T.: Sequential pattern mining using a bitmap representation. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining* (2002), pp. 429–435. URL: http://doi.acm.org/10.1145/775047.775109. 3

[BKL09] BIRD S., KLEIN E., LOPER E.: *Natural Language Processing with Python*, 1st ed. O'Reilly Media, Inc., 2009. 7

[BOH11] BOSTOCK M., OGIEVETSKY V., HEER J.: D^3: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2301–2309. URL: http://dx.doi.org/10.1109/TVCG.2011.185. 7

[BP07] BRANDES U., PICH C.: *Eigensolver Methods for Progressive Multidimensional Scaling of Large Data*. Springer, 2007, pp. 42–53. URL: http://dx.doi.org/10.1007/978-3-540-70904-6_6. 3

[EFN12] ENDERT A., FIAUX P., NORTH C.: Semantic interaction for visual text analytics. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (2012), pp. 473–482. URL: https://doi.org/10.1145/2207676.2207741, doi:10.1145/2207676.2207741. 10

[FP16] FEKETE J.-D., PRIMET R.: Progressive analytics: A computation paradigm for exploratory data analysis. *ArXiv e-prints* (July 2016). URL: http://arxiv.org/abs/1607.05162. 3, 10

[FPDS12] FISHER D., POPOV I., DRUCKER S., SCHRAEFEL M. C.: Trust me, I'm partially right: incremental visualization lets analysts explore large datasets faster. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (2012), pp. 1673–1682. URL: http://doi.acm.org/10.1145/2207676.2208294. 3, 4

[GKW14] GLUECK M., KHAN A., WIGDOR D. J.: Dive in!: Enabling progressive loading for real-time navigation of data visualizations. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (2014), ACM, pp. 561–570. URL: http://doi.acm.org/10.1145/2556288.2557195. 3

[HCPT05] HETZLER E. G., CROW V. L., PAYNE D. A., TURNER A. E.: Turning the bucket of text into a pipe. In *Proceedings of the IEEE Symposium on Information Visualization* (2005), IEEE Computer Society, pp. 89–94. URL: http://dx.doi.org/10.1109/INFOVIS.2005.34. 3, 4

[HG14] HUTTO C. J., GILBERT E.: Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the AAAI Conference on Weblogs and Social Media* (2014), pp. 216–225. 5

[HHW97] HELLERSTEIN J. M., HAAS P. J., WANG H. J.: Online aggregation. In *Proceedings of the ACM Conference on Management of Data* (1997), pp. 171–182. URL: http://doi.acm.org/10.1145/253260.253291. 3

[Hor99] HORVITZ E.: Principles of mixed-initiative user interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (1999), pp. 159–166. URL: https://doi.org/10.1145/302979.303030. 10

[Hor07] HORVITZ E. J.: Reflections on challenges and promises of mixed-initiative interaction. *AI Magazine 28*, 2 (2007), 3. URL: http://www.aaai.org/ojs/index.php/aimagazine/article/view/2036/1929. 10

[HR07] HEER J., ROBERTSON G.: Animated transitions in statistical data graphics. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1240–1247. URL: http://dx.doi.org/10.1109/TVCG.2007.70539. 5

[KAF*08] KEIM D., ANDRIENKO G., FEKETE J.-D., GÖRG C., KOHLHAMMER J., MÉLANCON G.: Visual analytics: Definition, process and challenges. In *Information Visualization - Human-Centered Issues and Perspectives*, no. 4950 in LNCS. Springer, Aug 2008, pp. 154–175. URL: https://link.springer.com/chapter/10.1007/978-3-540-70956-5_7. 1, 3

[KCL*17] KIM H., CHOO J., LEE C., LEE H., REDDY C., PARK H.: Pive: Per-iteration visualization environment for real-time interactions with dimension reduction and clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2017). URL: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14381. 1

[KKEM10] KEIM D. A., KOHLHAMMER J., ELLIS G., MANSMANN F.: *Mastering the Information Age: Solving Problems with Visual Analytics*. 2010. URL: http://www.vismaster.eu/wp-content/uploads/2010/11/VisMaster-book-lowres.pdf. 2

[KLTH12] KAPOOR A., LEE B., TAN D. S., HORVITZ E.: Performance and preferences: Interactive refinement of machine learning procedures. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2012). 10

[KMN*02] KANUNGO T., MOUNT D. M., NETANYAHU N. S., PIATKO C. D., SILVERMAN R., WU A. Y.: An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 7 (2002), 881–892. URL: http://dx.doi.org/10.1109/TPAMI.2002.1017616. 6

[LH14] LIU Z., HEER J.: The effects of interactive latency on exploratory visual analysis. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2122–2131. URL: http://dx.doi.org/10.1109/TVCG.2014.2346452. 1, 7

[MH08] MAATEN L. V. D., HINTON G.: Visualizing data using t-SNE. *Journal of Machine Learning Research 9*, Nov (2008), 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html. 5

[Mil68] MILLER R. B.: Response time in man-computer conversational transactions. In *Proceedings of the ACM Fall Joint Computer Conference* (1968), pp. 267–277. URL: http://doi.acm.org/10.1145/1476589.1476628. 1

[MPG*14] MÜHLBACHER T., PIRINGER H., GRATZL S., SEDLMAIR M., STREIT M.: Opening the black box: Strategies for increased user involvement in existing algorithm implementations. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1643–1652. URL: http://dx.doi.org/10.1109/TVCG.2014.2346578, doi:10.1109/TVCG.2014.2346578. 1, 3, 6, 9

[MvWvL99] MULDER J. D., VAN WIJK J. J., VAN LIERE R.: A survey of computational steering environments. *Future Generation Computer Systems 15*, 1 (Feb. 1999), 119–129. URL: http://dx.doi.org/10.1016/S0167-739X(98)00047-8. 3

[PLvdM*16] PEZZOTTI N., LELIEVELDT B., VAN DER MAATEN L., HOLLT T., EISEMANN E., VILANOVA A.: Approximated and user steerable tSNE for progressive visual analytics. *IEEE Transactions on Visualization and Computer Graphics* (2016), to appear. URL: http://dx.doi.org/10.1109/TVCG.2016.2570755. 1, 3

[PPM04] PEDERSEN T., PATWARDHAN S., MICHELIZZI J.: Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at HLT-NAACL 2004* (2004), Association for Computational Linguistics, pp. 38–41. 2

[PVG*11] PEDREGOSA F., VAROQUAUX G., GRAMFORT A., MICHEL V., THIRION B., GRISEL O., BLONDEL M., PRETTENHOFER P., WEISS R., DUBOURG V., VANDERPLAS J., PASSOS A., COURNAPEAU D., BRUCHER M., PERROT M., DUCHESNAY E.: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830. 7

[SASS15] SCHULZ H.-J., ANGELINI M., SANTUCCI G., SCHUMANN H.: An enhanced visualization process model for incremental visualization. *IEEE Transactions on Visualization and Computer Graphics 22*, 7 (July 2015), 1830–1842. URL: https://doi.org/10.1109/TVCG.2015.2462356. 3, 4, 10

[Shn84] SHNEIDERMAN B.: Response time and display rate in human performance with computers. *ACM Computing Surveys 16*, 3 (sep 1984), 265–285. URL: http://doi.acm.org/10.1145/2514.2517. 1

[SJ72] SPARCK JONES K.: A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation 28*, 1 (1972), 11–21. URL: http://dl.acm.org/citation.cfm?id=106765.106782. 5

[SPG14] STOLPER C. D., PERER A., GOTZ D.: Progressive visual analytics: User-driven visual exploration of in-progress analytics. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1653–1662. URL: http://dx.doi.org/10.1109/TVCG.2014.2346574. 1, 3, 4

[Tho05] THOMAS J. J.: *Illuminating the path: The research and development agenda for visual analytics*. IEEE Computer Society, 2005. 1

[TSSW10] TUMASJAN A., SPRENGER T. O., SANDNER P. G., WELPE I. M.: Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of the AAAI Conference on Weblogs and Social Media* (2010), vol. 10, pp. 178–185. 2

[Tuk77] TUKEY J. W.: *Exploratory data analysis*. Addison-Wesley series in behavioral science: quantitative methods. Addison-Wesley, Reading (Mass.), 1977. 1

[WM04] WILLIAMS M., MUNZNER T.: Steerable, progressive multidimensional scaling. In *Proceedings of the IEEE Symposium on Information Visualization* (2004), pp. 57–64. URL: http://dx.doi.org/10.1109/INFVIS.2004.60, doi:10.1109/INFVIS.2004.60. 3

[YGRP12] YANG X., GHOTING A., RUAN Y., PARTHASARATHY S.: A framework for summarizing and analyzing Twitter feeds. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining* (2012), pp. 370–378. URL: http://doi.acm.org/10.1145/2339530.2339591. 2

[ZGC*16] ZGRAGGEN E., GALAKATOS A., CROTTY A., FEKETE J.-D., KRASKA T.: How progressive visualizations affect exploratory analysis. *IEEE Transactions on Visualization and Computer Graphics* (2016), to appear. URL: http://dx.doi.org/10.1109/TVCG.2016.2607714. 1, 3, 4, 7, 10