Visual Analytics of Temporal Event Sequences in News Streams

Dissertation zur Erlangung des akademischen Grades
des Doktors der Naturwissenschaften an der Universität Konstanz
im Fachbereich Informatik und Informationswissenschaft

vorgelegt von

Miloš Krstajić

Universität Konstanz

Tag der mündlichen Prüfung: 18. Juni 2014

Referenten:

Prof. Dr. Daniel A. Keim, Universität Konstanz

Prof. Dr. William Ribarsky, University of North Carolina at Charlotte

Abstract

Finding new ways of extracting and analyzing useful information from exploding volumes of unstructured and semi-structured text sources has become one of the greatest challenges in the era of big data. After new technologies have enabled efficient solutions for collecting and storing these data, the next step in computer science research is to develop scalable approaches for efficient analysis of dynamics in text streams. This dissertation addresses this challenge by examining how visual analytics can help the users gain new insights from systems for explorative analysis of events in text streams that are more efficient and easier to use. My work revolves around the concept of streaming visual analytics, whose goal is to combine resource constraints of the computer and time constraints of the user to provide more scalable tools. I identify challenges in the user, data and visualization domain, discuss open issues and derive design considerations to help practitioners in developing future systems for incremental data. Based on this approach, I describe novel visual analytics methods for detection and exploration of events in news streams: CloudLines, a compact overview visualization for events in multiple event sequences in limited space, and Story Tracker, a visual analytics system for exploration of news story development and their complex properties. Novel experimental visualizations are introduced to demonstrate the applicability of the approach in real time monitoring scenarios. I describe how the streaming visualization concepts pervade my work and outline directions for future research.

Zusammenfassung

Eine der größten Herausforderungen des Big Data Zeitalters ist die Extraktion und Analyse von nützlichen Informationen aus großen und wachsenden Datenvolumina. Neu entwickelte Technologien ermöglichen effizientes Abrufen und Speichern von Textquellen. Die daraus nachfolgende Herausforderung fr die Informatik besteht darin, skalierbare Methoden zur Analyse der Dynamik von Textdatenströmen zu entwickeln. Diese Dissertation bietet Lösungen zu diesen Herausforderungen und untersucht, wie Visual Analytics die Benutzer unterstützen kann, neue Einsichten mit Hilfe von Systemen zur explorativen Analyse von Ereignissen in Textströmen zu gewinnen, wobei diese effizient und intuitiv zu benutzen sind. Meine Arbeit stellt das Konzept der "Streaming Visual Analytics" vor, dessen Ziel es ist, ressourcenbedingte Einschränkungen des Computers und Zeitvorgaben von Benutzern zu verbinden, um skalierbarere Werkzeuge zur Analyse von Textströmen bereitzustellen. Dafür werden Herausforderungen in Hinsicht auf Benutzer, Daten und Visualisierungen für Analysewerkzeuge identifiziert sowie offene Fragestellungen in diesem Forschungsthema diskutiert. Des Weiteren stelle ich Gestaltungsprinzipien bereit, um Forschern bei der Entwicklung neuer Systeme zur inkrementellen Datenanalyse zu helfen. Basierend auf diesem Ansatz beschreibe ich neue Visual Analytics Methoden zur Erkennung und Exploration von Ereignissen in Nachrichtenströmen: CloudLines, eine kompakte Visualisierung von Ereignissen auf beschränktem Raum, die in mehrere parallele Sequenzen gegliedert werden, oder Story Tracker, ein Visual Analytics System zur Exploration der Entwicklung von Nachrichtengeschichten sowie ihrer komplexen Eigenschaften. Des Weiteren werden innovative Visualisierungen vorgestellt, mit deren Hilfe die Praktikabilität dieses Ansatzes in Echtzeit-Überwachungsszenarien demonstriert wird. Ich beschreibe, inwiefern Streaming Visualisierung diese Arbeit motiviert sowie strukturiert und schneide weitere Forschungsmöglichkeiten in diesem Bereich an.

To my mom and dad

Acknowledgements

I would like to thank my advisor Prof. Dr. Daniel A. Keim for the continuous support throughout my PhD. He gave me scientific freedom to choose and pursue my topic and guided me by giving valuable advice when it was needed. He also showed me that it is possible to raise a family and be tremendously successful in research.

I would like to thank Dr. Florian Mansmann for fruitful collaboration and joyful research discussions; for being open and one of the smartest people I met; for being young but calm; and - for being a friend. I would like to thank Dr. Enrico Bertini for reminding me that we are human beings; for passionate research sessions and our fights; for reminding me about the importance of the family. This dissertation would not have been possible without their daily advice and passion for doing research on visualization of incremental data together.

I would like to thank Prof. Dr. Bill Ribarsky for the external collaboration and reviewing my dissertation, as well as Prof. Dr. Oliver Deussen and Prof. Dr. Tobias Schreck for giving valuable advice, support and being my committee members. I would also like to thank Prof. Dr. Dietmar Saupe and the professors from the DFG Graduate School "Explorative Analysis and Visualization of Large Information Spaces".

I would like to thank Dr. Christian Rohrdantz, Dr. Andreas Stoffel, Dr. Peter Bak, Dongning Luo, and all my colleagues from the Data Analysis and Visualization Group, as well as students and external collaborators, for sharing passion for visual analytics and doing research together.

A special thanks goes to Dr. Andrada Tatu for sharing and solving problems; for being there after work; for helping me tremendously improve my Deutsch; Svenja Simon for sharing our "girls" office and having a great time; Slava Kisilevich for surviving all of the typical research problems of young scientists together; Matthias Schaefer for being a friend from the first day I met him and having a great Swabian sense of humour. I would also like to thank Juri Buchmüller, Florian Stoffel and the rest of the tech support crew for providing an excellent service to our research group.

Finally, my biggest gratitude goes to Rada, for her love, support and understanding she showed throughout our life together; to my son Ivo, for being a big source of motivation and joy; to my mom, dad and my brother for believing in me; for giving unconditional support and being there in the good times and the bad times. A big thank you goes to all of my friends and other family members who constantly reminded me that there is life out there and making me love Belgrade even more.

Contents

1 Introduction

Contents

Crisis in Ukraine.
Gay Marriage Protests In France Draw Thousands.
Protests in Greece.
A bomb in Pakistan.
Bayern Munich wins the Champions League.
Right-wing parties win Norway election.

In the era of big data, we are overwhelmed by news about important global and local events in politics, business, sports and entertainment. A vast amount of news articles is published every day all over the world covering these events from different angles. Global media agencies and thousands of news portals continuously produce new content, which quickly replaces the old. Although the context of the past events can very often help in interpreting new events, the volume and speed of these information streams make it more difficult to understand what is really going on and relate the current events to the events from the past. Simultaneously, a whole new space of user-generated data has been created with the increasing popularity of blogs, user comments, reviews, twitter and other channels where people can give feedback on different topics. One-way media street has become a two-way media information highway with the traditional readers now being the content creators and not just passive consumers of the generated information. Large streams of textual data are fast, complex, and contain semi-structured and unstructured information.

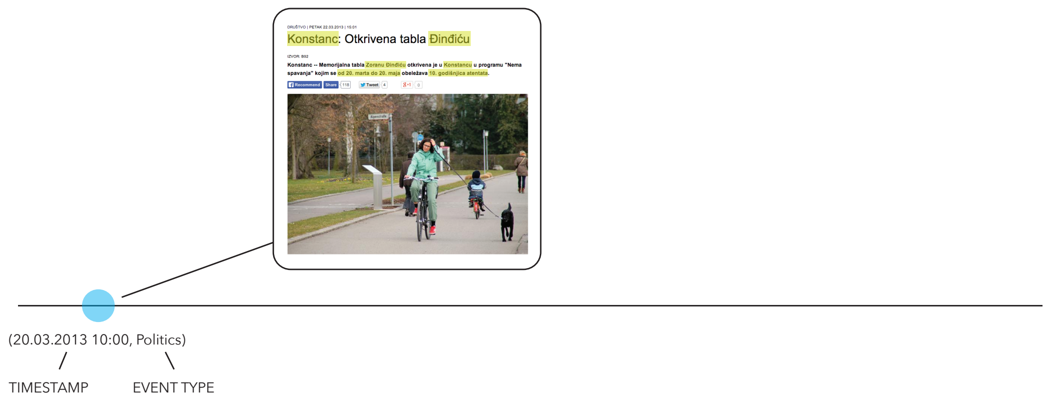
The volume of unstructured and semi-structured textual data is exploding, while our human resources and time available to process this type of content are remaining rather constant. Therefore, it is becoming increasingly harder to identify, analyse, explore and ultimately understand the patterns that appear in these rich sources of information.

Breaking news providers and so called *backchannels* together create an enormous information space with huge potential for new developments in computer science, social sciences and business. Companies that monitor public opinion offer services that provide qualitative information about popularity of political parties, their leaders and other organizations. They can help their customers to make smarter decisions on the new policies they are developing and allow them to react quickly to scandals or other unexpected events. Advertising and public relations agencies create campaigns for brands and companies and need to be able to track the effectiveness of their work both in the mainstream media channels and within the customer communities. Being able to analyze in detail the media coverage around a new product launch or quickly respond to a crisis created by a defective product or poor service will give the advantage to those companies who are efficiently using the information that is burried in the huge volumes of online text streams. Government bodies need to respond in emergency situations and send help to areas affected by unexpected disasters.

All above-mentioned businesses need tools and algorithms that would help them extract and analyze relevant information from these data. Although a lot of effort has been put into developing efficient data storage and management solutions, analytical tools that would help the user understand these vast amounts of data are lagging behind.

1.1 Thesis Problem and Approach

The dissertation focuses on the central problem of how to design *visual analytics* tools that support analysis of interesting temporal events in text streams. I have used the visual analytics approach, which combines automated computational methods with interactive visualization techniques to facilitate processing, analysis and understanding of the data and has been defined as ”the science of analytical reasoning facilitated by visual interactive interfaces” [166]. The main concept of visual analytics is to combine automated computational algorithms with visual representations and interaction techniques, leveraging advantages of computers and humans at different stages of the analytical process. It is a field that naturally expanded from *information visualization*, which provided the foundations for finding appropriate visualization methods. In particular, I examined how this approach can be used to make sense of large unstructured data such as text streams, focusing on specific properties and temporal events that appear in these streams. I investigated which technical and analytical challenges exist during different stages of the visual analytics process of transforming raw data to insight.


DRUŠTVO | PETAK 22.03.2013 | 15:01

Konstanc: Otkrivena tabla Đinđiću

IZVOR: B92

Konstanc -- Memorijalna tabla Zoranu Đinđiću otkrivena je u Konstancu u programu "Nema spavanja" kojim se od 20. marta do 20. maja obeležava 10. godišnjica atentata.

Figure 1.1: Definition of an event in a news stream: A time-stamped record with additional information: event type, unstructured text and metadata

In my thesis, a *temporal event*, or an *event* is defined as a time-stamped document, a record consisting of a *timestamp* (June 28, 2013, 14:10 CET) and additonal information - *event type*, *text* and *metadata* (Figure 1.1). The *event* *type* depends on the data that I am working with - for example, it can be a name of a politician that is mentioned in a news article, or the name of the website that published the article[1]. The *text* is the unstructured or semi-structured textual content of the record - full text of the document or a tweet, while the *metadata* is any additional information about the document, which can be the location of the news publisher, the location mentioned in the document, a set of keywords, the language of the document, the original URL or a link to an image.

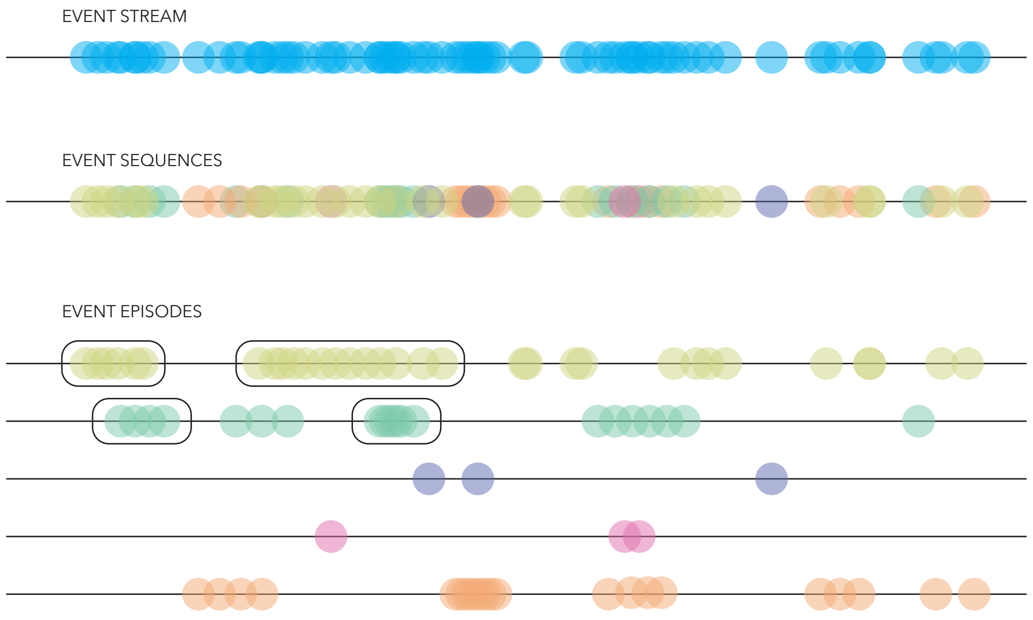


Figure 1.2: Event stream, event sequences and event episodes

The challenges arise when many of these events appear together in an *event stream*. Events of the same event type create *event sequences*, and those events that are similar in some way and temporally close in time create *event episodes*. These event episodes can be, for example, a group of articles (from different sources) reporting on the same real-world event (Figure 1.2).

[1]Please note the difference between my definition of an event and the real-world event. According to this definition, it is possible (and expected) that several events in the text stream are related to the same real-world event - for example, if these events are created by different sources.

Following these definitions, the events, episodes and sequences can be found in many other domains - we can easily think of network traffic logs that generate such data in billions, financial transactions that are very often recorded with additional textual information, or personal logs that keep track of someone's training history.

My work was initially inspired by the discussions I had with various journalists, and the researchers from the European Commission's Joint Research Centre. They have been developing an extensive news aggregator service for many years, Europe Media Monitor, which is still actively maintained and upgraded[2]. Although many years of research in the field of information visualization has brought a myriad of techniques suitable for visualizing different data types, I realized that there is a research gap in providing an overview of multiple long event sequences at once while being able to directly interact with each event. To address this problem, I developed *CloudLines*, an interactive visualization method for detection and analysis of event episodes in multiple event sequences. This initial problem has ignited further work in three domains - *text*, *time* and *streams*, which represent the research problem space of my thesis (Figure 1.3).

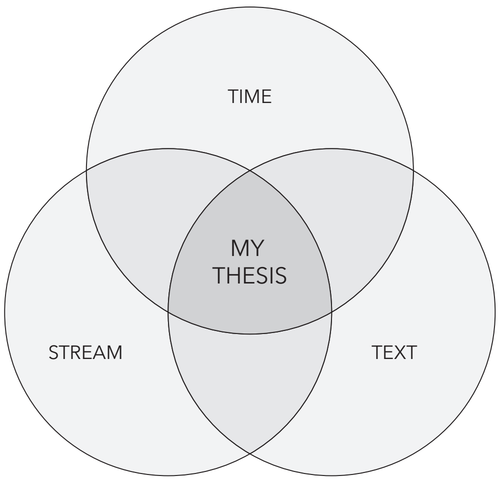


Figure 1.3: Thesis problem space: Challenges in visual analytics of temporal event sequences in text streams appear in three domains: text, time and streams

[2]http://emm.newsbrief.eu/

These three domains are often interconnected and solving a specific analytical task means that a research challenge exists in two domains at the same time (or even all of them), which significantly increases the degree of the problem complexity. An illustrating example is *the analysis of news stories*. Beginning with the **text** domain, in the area of information retrieval and text mining, there is an extensive research on document summarization and text clustering, which aims at providing summaries of document corpora and generating meaningful groups of similar documents, to help the users in dealing with abundance of textual information. Identification of *important* stories requires sophisticated relevance measures for cluster ranking. Some news stories can be related to each other, so we need methods to understand the *relationships* between the clusters. All these tasks become much more difficult once we add the **temporal** dimension to the problem. How are the stories *evolving* and how can we relate the current events to the past? How can we detect when stories *split* or *merge* over time? Finally, the **streaming** aspect deals with the fact that the text collections are constantly growing. It is computationally expensive to process the whole dataset every time a new document is added, and our visual representations could change in a way that would lead to significant loss of context. This means that we need efficient algorithms that will process, transform and visualize data effectively.

To address these problems, I have reviewed relevant prior work in information visualization, visual analytics, text mining and data stream management to develop guidelines and recommendations for visual analytics for streaming data. I have also examined several well-known information visualization methods and observed which visual variables can change and how. I described how these changes are related to the attribute and structural changes that can occur in the data stream. Based on the literature review and my experience in designing analytic tools for dynamic data, I conceptualized a research framework for streaming visual analytics, which is a first step towards a formal theoretical model.

This approach was maturing in parallel with the development of novel visual analytics methods for detection and exploration of events in news streams.

I designed *CloudLines*, a compact visualization for events in multiple event sequences in limited space that uses kernel density estimation to identify short intervals with a lot of events. I developed lens and timeline distortion as interaction techniques for CloudLines, as well as decay and cut-off functions to remove irrelevant events and improve performance.

Story Tracker is a visual analytics framework for incremental analysis of development of news stories, which can split and merge over time. It allows the user to steer the text clustering algorithms and refine the results at every stage of the data transformation and visualization processes. Text clustering algorithms extract stories from online news streams in consecutive time windows and identify similar stories from the past. The stories are displayed in a visualization, which (1) sorts the stories by minimizing clutter and overlap from edge crossings, (2) shows their temporal characteristics in different time frames with different levels of detail, and (3) allows incremental updates of the display without recalculating the past data. Stories can be interactively filtered by their duration and connectivity in order to be explored in full detail. Two use cases with real news data demonstrate the capabilities of the system for detailed dynamic text stream exploration.

A general comparison between the two approaches shows important conceptual differences (Table 1.1). First of all, CloudLines is used for event data whose type is already known, while Story Tracker is used for event data whose type is unknown. Next, CloudLines processes event sequences sequentially type by type, while Story Tracker processes all event data incrementally in batches in order to find similarities between events and dynamically detect new event types. In CloudLines, events are the basic visual objects, which create event episode visual aggregates, while the basic visual object in Story Tracker is a daily cluster, which represents a group of similar events in one batches. These daily clusters are connected to create story visual aggregates. Finally, the main interaction techniques in CloudLines are lens magnification, timeline distortion, selection and details on demand, while Story Tracker uses advanced filtering, hovering, highlighting, clustering parameter adjustment, reordering and details on demand.

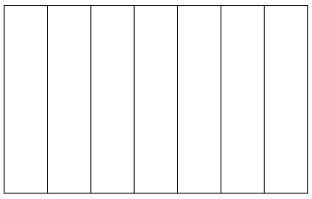
	CLOUDLINES	**STORY TRACKER**
	Compact Display of Event Episodes in Multiple Event Sequences	Incremental Visual Text Analytics of News Story Development
		
Event Type	PREDEFINED	UNKNOWN
Event Processing	SEQUENTIAL	PARALLEL
Visual Data Abstraction	EVENTS → EVENT EPISODES	DAILY CLUSTERS → STORIES
Visualization	EVENTS (NEWS ARTICLES)	STORIES
Interaction	LENS MAGNIFICATION DISTORTION SELECTION DETAILS ON DEMAND	FILTERING HIGHLIGHTING PARAMETER ADJUSTMENT REORDERING DETAILS ON DEMAND

Table 1.1: Comparison of the approaches: CloudLines vs Story Tracker

A special case for streaming visual analytics is real-time monitoring of critical issues in text streams. I developed several methods for detection and visualization of events in real-time.

1.2 Thesis Contributions

The major contributions of my thesis are:

1. Streaming visual analytics research framework, which describes design considerations for developing visual analysis tools suitable for handling incremental data sources, and identifies challenges and issues in the user, data and visualization problem space;

2. CloudLines, a novel visualization method for analysis of temporal events

in multiple sequences in limited space, which is coupled with interaction techniques for detailed exploration of events in data streams.

3. Story Tracker, a visual analytics system for analysis of text streams, which combines text clustering algorithms with incremental visualization to create a coherent analytical environment for analysis of news story development.

4. A set of experimental visual analytics methods for text streams, which demonstrate how streaming visualization techniques and event detection algorithms can be applied on text streams in real-time monitoring scenarios.

1.3 Thesis Outline

Chapter 2 introduces the streaming visual analytics framework by discussing challenges and issues in user, data and visualization domains, followed by design considerations and principles for visual analytics systems for streaming data.
Chapter 3 covers background and related work in the area of visual analytics for large text corpora, text mining and visualization of temporal data and data streams.
Chapter 4 describes interactive exploration of event episodes in news streams with *CloudLines*, an interactive compact overview visualization for multiple long event sequences in limited space.
Chapter 5 describes *Story Tracker*, a novel visual analytics framework for exploration of news story development.
Chapter 6 focuses on scenarios where the response times of the systems should be immediate.
Chapter 7 concludes the dissertation by summarizing the contributions, discussing limitations and future work for visual analysis of streaming data.

Parts of this thesis have been published in the following publications (ordered by their appearance in the thesis):

1. Milos Krstajic and Daniel A. Keim. Visualization of Streaming Data: Observing Change and Context in Information Visualization Techniques. Big Data Visualization Workshop at the 2013 IEEE International Conference on Big Data, 2013.

2. Christian Rohrdantz, Daniela Oelke, Milos Krstajic and Fabian Fischer. Real-Time Visualization of Streaming Text Data: Tasks and Challenges (Best Paper Award). Workshop on Interactive Visual Text Analytics for Decision-Making at the IEEE VisWeek 2011, 2011.

3. Daniel A. Keim, Milos Krstajic, Christian Rohrdantz and Tobias Schreck. Real-Time Visual Analytics for Text Streams. IEEE Computer 46(7): 47-55, 2013.

4. Milos Krstajic, Enrico Bertini and Daniel Keim. CloudLines: Compact Display of Event Episodes in Multiple Time-Series. IEEE Transactions on Visualization and Computer Graphics, 17:2432-2439, 2011.

5. Milos Krstajic, Mohammad Najm-Araghi, Florian Mansmann and Daniel A. Keim. Story Tracker: Incremental Visual Text Analytics of News Story Development. Information Visualization, SAGE, 12(3-4):308-323, 2013.

6. Milos Krstajic, Mohammad Najm-Araghi, Florian Mansmann and Daniel A. Keim. Incremental Visual Text Analytics of News Story Development (Best Paper Award). SPIE 2012 Conference on Visualization and Data Analysis, 2012.

7. Milos Krstajic, Florian Mansmann, Andreas Stoffel, Martin Atkinson, Daniel A. Keim. Processing Online News Streams for Large-Scale Semantic Analysis. ICDE 2010, 1st International Workshop on Data Engineering meets the Semantic Web (DESWeb), 2010.

8. Milos Krstajic, Enrico Bertini, Florian Mansmann and Daniel A. Keim. Visual analysis of news streams with article threads. StreamKDD '10: Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques, ACM KDD 2010, 2010.

9. Florian Mansmann, Milos Krstajic, Fabian Fischer and Enrico Bertini. StreamSqueeze: A Dynamic Stream Visualization for Monitoring of Event Data. SPIE 2012 Conference on Visualization and Data Analysis (VDA '12), 2012.

Other publications that were published during my work on the dissertation that indirectly influenced it are:

- Dongning Luo, Jing Yang, Milos Krstajic, William Ribarsky, Daniel A. Keim. EventRiver: Visually Exploring Text Collections with Temporal References. IEEE Transactions on Visualization and Computer Graphics, Vol. 18, No. 1, pp. 93-105, 2012.

- Daniel A. Keim, Leishi Zhang, Milos Krstajic, Svenja Simon. Solving Problems with Visual Analytics: Challenges and Applications. Journal of Multimedia Processing and Technologies, Special Issue on Theory and Application of Visual Analytics, Vol. 3, No. 1, pp. 1-11, 2012.

- Milos Krstajic, Christian Rohrdantz, Michael Hund, Andreas Weiler. Getting There First: Real-Time Detection of Real-World Incidents on Twitter. 2nd IEEE Workshop on Interactive Visual Text Analytics Task-Driven Analysis of Social Media, October 2012.

- Christian Rohrdantz, Milos Krstajic, Mennatallah El Assady, Daniel A. Keim. What is Going On? How Twitter and Online News Can Work in Synergy to Increase Situational Awareness. 2nd IEEE Workshop on Interactive Visual Text Analytics Task-Driven Analysis of Social Media, October 2012.

- Michael Behrisch, Milos Krstajic, Tobias Schreck, Daniel A. Keim. The News Auditor: Visual Exploration of Clusters of Stories. Eurographics, pp. 61-65, 2012, Eurographics Association.

- Milos Krstajic, Peter Bak, Daniela Oelke, Martin Atkinson, William Ribarsky, Daniel A. Keim. Applied visual exploration on real-time news feeds using polarity and geo-spatial analysis. 6. International Conference on Web Information Systems and Technologies (WEBIST), 2010.

- Slava Kisilevich, Milos Krstajic, Daniel A. Keim, Natalia Andrienko, Gennady Andrienko. Event-based analysis of people's activities and behavior using Flickr and Panoramio geo-tagged photo collections. 2. International Symposium on Visual Analytics (IV), 2010.

2 Towards Streaming Visual Analytics

Contents

An extensive body of research exists in several areas that are relevant for visual analytics of dynamically changing data. However, there are very few integrated approaches that treat the data as streams (or incremental data sources in general) at every step of the visual analytics process. One of the reasons is that the research is at a very early stage and there are still many open issues in each domain separately. Although solving a problem step by step is a common approach in the incremental science, the lack of a theory that would serve as a foundation for the invention of the new methods and a roadmap for defining these important problems can sometimes lead to techniques, methods and approaches that actually can not be deployed in a real-world scenario. A systematic overview of the most important challenges does not exist and the researchers are very often hitting the wall once they start applying the existing knowledge about (static) visual data analysis to dynamic incremental sources. The goal of this chapter is to make first steps in bridging this gap.

This chapter introduces the research framework for streaming visual analytics systems. The rest of this chapter is structured as follows: section 2.1 describes the background of the problem and presents the tasks and applications related to analysis of streaming data. Section 2.2 introduces the terminology and my approach for streaming visual analytics. Section 2.3 identify challenges derived from reviewing existing methods. In section 2.4, i summarize and describe the properties of incremental visualizations, while section 2.5 summarizes the design considerations. The classification of information visualization methods based on their incremental scalability is provided is presented in section 2.6. Section 2.7 concludes the chapter.

This chapter is partially based on the following publications:

Milos Krstajic, Florian Mansmann, Oliver Deussen and Daniel Keim. From Static to Dynamic: Towards Streaming Visual Analytics[1]

Milos Krstajic and Daniel A. Keim. Visualization of Streaming Data: Observing Change and Context in Information Visualization Techniques. Big Data Visualization Workshop at the 2013 IEEE International Conference on Big Data, 2013. [106][2]

Christian Rohrdantz, Daniela Oelke, Milos Krstajic and Fabian Fischer. Real-Time Visualization of Streaming Text Data: Tasks and Challenges (Best Paper Award). Workshop on Interactive Visual Text Analytics for Decision-Making at the IEEE VisWeek 2011, 2011. [142][3]

Daniel A. Keim, Milos Krstajic, Christian Rohrdantz and Tobias Schreck. Real-Time Visual Analytics for Text Streams. IEEE Computer 46(7): 47-55, 2013. [96][4]

[1]in preparation

[2]The concept for the paper was created by both authors. I did the research and wrote the paper and Daniel Keim supervised the execution and gave advice.

[3]All authors contributed to the paper in the research discussions by reiterating the paper structure, outline and important points. The sections were divided together, and the order is based on the effort put into writing. The final version was proofread by all authors. The parts not written by me are indented in the text

[4]The order of the authors is alphabetical and all authors contributed to the paper in

2.1 Background

We live in a dynamic environment of big data, where analyzing, visualizing and interacting with dynamic information is key to solve many real-world tasks, which are related not only to emergency management, where immediate response is needed, but also in day-to-day operations in areas such as finance, network security, news analysis, and social networking that rely on evolving data. Traditional methods in information visualization, visual analytics and knowledge discovery rely on static data sets and are not bounded by strict time and resource constraints. In a conventional setting, the user has a large dataset of historical data, which is stored in a database, with enough time for data processing and exploration. However, in many applications the environment is much more dynamic - the dataset is growing rapidly and cannot be stored in a traditional way, the offline data mining algorithms are computationally too expensive to be run every time the data set is changed, and the time available for performing exploration tasks is short.

Although Thomas and Cook [166] defined the purpose of visual analytics tools to "provide timely, defensible, and understandable assessments", little research has dealt with the *timely* aspect. Likewise, Chen identified the "paradigm shift from structures to dynamics" as one of the top 10 unsolved problems in the information visualization field in 2005 [37] and it is explicitly mentioned as a data challenge in chapter 9 of the book "Mastering the Information Age - Solving Problems with Visual Analytics" [95].

But why is visual analytics of streaming data difficult? Mansmann et al. [120] argue that this is due to the technical intricacies of streams, which require expertise not only in the visualization domain, but also in data management. The research on data streams has been evolving independently in data management, data mining and visualization community to improve data handling, develop online data mining methods and integrate visual representation, user tasks and interaction. In the essence, an interdisciplinary approach is needed to master the data stream analysis challenges.

research meetings. The paper was proofread by all authors.

Although there has been a certain increase in research efforts in the visual analytics community to develop systems for real-time analysis in emergency response usage scenarios, there is a lack of systematization of issues that arise when the underlying data set is constantly evolving.

The goal of this chapter is to identify key issues related to visual analysis of streaming data and their complex properties. The focus is on the aspects of the data streaming environment related to: 1) data volume and the rate of change, 2) notion of time in data streams and visualizations, and 3) the role of the user. I analyze how these different aspects relate to challenges in developing functional visual analytics systems, discuss the insights from existing approaches and identify design considerations for visual analysis for streaming data.

These considerations are reflected in the development of methods and systems for visual analysis of text streams which were developed in this thesis and are presented in the later chapters.

2.1.1 Tasks and Applications

Tasks and analysis that require data from incremental data sources are highly dependent on solving challenges in real time data processing. In the era of big data, global competition requires shorter response times and faster decision making. Therefore, the tasks are closely related to solving technical challenges, as there are many real-time applications where users need to get insights into data streams immediately to facilitate decision making. The implementations of such applications and their visualization techniques are strongly influenced by the overall tasks and analysis goals.

> According to Thomas et al. [166] three significant problems are: 1) to provide situational awareness; 2) to show changes; and 3) fuse different data sources. This leads to the main tasks for the visual analysis of streaming text data, which are monitoring, decision making, change and trend detection, event tracking, historical retrieval and exploration of data items to eventually achieve situational awareness [56].

When analyzing data streams, the analysts want to identify trends, detect changes and unusual patterns at reasonable level of detail [72]. In order to enable aggressive analysis and management of information and knowledge from relevant data, real time streaming analytical systems [136] should:

- respond in real time to events and changing requirements
- continuously analyze data at rates that are orders of magnitude greater than existing systems
- adapt rapidly to changing data forms and types
- manage high availability, heterogeneity, and distribution for the new stream paradigm
- provide security and information confidentiality for shared information

2.2 Streaming Visual Analytics: Research Framework

Kleinberg discusses two aspects of analyzing data streams: first, the data stream volume and rate is too large to be stored, which creates new requirements for efficiency and scalability [103]. In the data streaming community, the first aspect combined with data mining goals creates two divergent objectives [61]: 1) The analysis should produce comprehensive and exact results and detect changes in the data as soon as possible; 2) Resource limitations allow only to perform the analysis on an approximation of the stream (e.g., *samples* or *sketches*) or a window (i.e., a finite subset of the stream). The second aspect focuses on temporal, dynamic processes that exist in the information streams. My approach combines these two aspects by proposing development of systems from the user perspective - the approximation of the stream should be performed when needed, the archived information should be provided when required by user tasks, and the dynamics should be explored when possible.

2.2.1 Terminology and Definitions

Before presenting the streaming visual analytics approach, the relevant terms are defined and revised in this section. Afterwards, the challenges related to data handling, visualization and user interaction are identified, followed by the summary of incremental visualization properties. Next, the adaptability of several well-known information visualization techniques to incremental data sources is analyzed. Finally, the recommendations and open issues for future streaming visual analytics systems are provided.

Definition 2.1 *Streaming Visual Analytics integrates streaming visualization with automated analysis methods for streaming data to support the analyst in a timely fashion.*

This definition fulfills the requirement from *The R&D Agenda for Visual Analytics*: "Visual analytics must facilitate high-quality human judgment with a limited investment of the analysts' time" [166].

Streaming data model

A data stream is an ordered sequence of items that arrives in timely order. Unlike data in traditional static databases, data streams are ordered sequences of items. They are continuous, unbounded, and usually come with high speed and have a data distribution that often changes with time [71]. Storing and processing all the historical data is not efficient and would practically require infinite storage and running time. In my work, I consider data streams that rely on a model described in [14]: (a) *"The data elements in the stream arrive online."* (b) *"The system has no control over the order in which data elements arrive to be processed, either within a data stream or across data streams."* (c) *"Data streams are potentially unbounded in size."* (d) *"Once an element from a data stream has been processed it is discarded or archived it cannot be retrieved easily unless it is explicitly stored in memory, which typically is small relative to the size of the data streams."*

Text stream

In the literature, *text stream* is used to describe: (a) a time-stamped, temporally ordered text collection, or (b) a data stream containing unstructured or semi-structured text. The first definition does not imply that the data analysis and visualization algorithms work with a constantly evolving dataset that has to be processed online.

Streaming visualization

Definition 2.2 *Streaming visualization is an incremental visualization technique that uses the streaming data model as the underlying data source.*

However, in the visualization community, the term *streaming visualization* has been used in different contexts, which sometimes do not fit well with the definition of a streaming data model. Very often, this term describes a visualization of a (time-stamped) dataset that originated from a data stream, and not necessarily a visualization that can deal with the incremental nature of the stream.

Incremental vs dynamic visualization

The terms that are closely related to streaming and have been in use for a long time are *dynamic* and *incremental*. Dynamic visualization (or representation) has been used to describe a change in the data. Sometimes, dynamic representation refers to motion/animation of a static dataset that contains time-oriented data."Static representations visualize time-oriented data in still images (i.e., representations that do not change automatically over time). In contrast to that, dynamic representations utilize the physical dimension time to convey the time dependency of the data (i.e., representations that change automatically over time such as slide shows or animations). The presence or absence of interaction facilities has no influence on whether a visualization approach is categorized as static or dynamic." [5]. In graph visualization, dynamic visualization of an evolving social network can be actually pre-computed offline and than artificially updated to show the change. In some cases that appeared in cognitive science research on information visualization [80], any visualization that can be dynamically queried [153] is referred to as ”dynamic”.

Incremental visualization is much closer to *streaming visualization* as it refers to any visualization method that can be updated without recalculating the layout of the whole visualization and display objects. It does not necessarily include the requirement of forgetting the past data, which is implied by the definition of the data stream.

Mansmann et al. [119] have used the term *dynamic visual analytics*, which is defined as “the process of integrating knowledge discovery and interactive visual interfaces to facilitate data stream analysis and provide situational awareness in real-time.”

2.2.2 Approach

The approach follows the visual analytics process diagram by Keim et al. [90], shown in Figure 2.1, which has two fundamental assumptions:

1. the dataset is static
2. the user has sufficient (*unlimited*) time for analysis

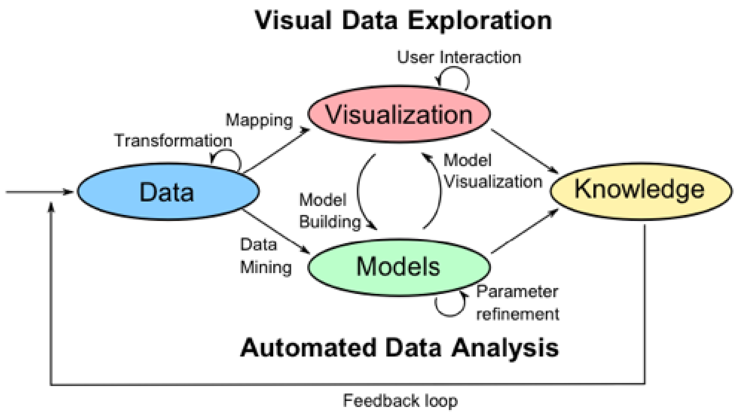


Figure 2.1: The visual analytics process model by Keim et al. [90]

The visual analytics process allows the user to manipulate the data at each stage: transform, filter, combine the data, change the parameters and refine the model, set the visualization parameters, or start from scratch with a different dataset. Although this model can formally fit to streaming data, it is not clear in which order would the components execute when the data changes and how would this affect the flow of the information from the data source to the knowledge component once new data is injected. Moreover, the user has limited time for analysis, which means that rebuilding the model and repeated computation of the visualization layout with the whole data would be too expensive.

Streaming visual analytics identifies three processes that are running in parallel, with strong interaction: 1) data stream itself which is continuously changing; 2) data processing (which includes data transformation, management, mining and visualization); and 3) user interaction (exploration, filtering, etc), as shown in Figure 2.2.

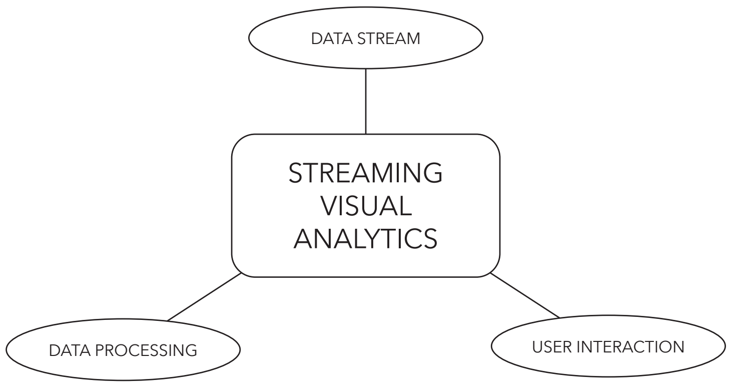


Figure 2.2: Streaming Visual Analytics depends on three processes that run in parallel: data stream, data processing and user interaction, each of which is has its own running times. The user interaction time should be defined by the application and ultimately determine the parameters of the other two processes.

Each of these processes is characterized by its own *speed*. Data stream is any incremental source and its speed is defined by the change in the data volume over time. The data processing consists of several generic modules, which can be then divided into submodules, e.g. data transformation, data filtering, clustering, visualization etc. Speed of each of the used submodules is given implicitly by the amount of time needed for the submodule execution. The third process represents the user and includes all the exploration tasks that occur during data analysis. In this case, the speed is the change of the "amount" of interactivity over time.

The *user interaction time* (UI time), which is the time necessary to complete the analysis tasks that are defined by the application, determines the parameters of the first two processes.

An example of a task where the UI time is *short* are monitoring tasks - in these tasks the user is usually not performing complex interaction. He is interested in detecting unexpected behaviour in the stream and the visualized data can be updated as soon as it becomes available. The availability depends on the amount of data in the stream, as well as the time needed to transform the raw data into the desired form that will be shown to the user.

An example of a task with a long UI time is the exploration of development of news stories. After identifying stories of interest, the user needs to find answers to *five Ws*[5]: *what* happened, *who* was involved, *why*, *when* and *where* did it happen. This would require a combination of different techniques for navigating in the topic space, filtering of information and ultimately reading text, which are complex and long tasks. At the same time, new events appear in the data stream and need to be processed. The streaming visual analytics reasoning now takes place - it is of utmost importance to understand how big the new data is, what data processing steps need to be taken and how long will it take to compute them, and how will the new data be shown to the user in an unobtrusive and informative manner.

MONITORING
EXPLORATIVE ANALYSIS
short UI time
long UI time

Figure 2.3: Monitoring applications usually require short user interaction times, while explorative analysis tasks usually require long user interaction times.

These two illustrative examples represent two opposite cases on the UI timeline. In between exist plethora of user tasks of different complexity. It is

[5]These are typical information gathering questions used by journalists, researchers and police investigators.

obviously not possible to estimate the exact time needed to solve each task and the proposed approach does not aim to do so. The goal is to identify important processes that the researcher needs to take into account when developing visual analytics systems for data streams and it goes hand in hand with the existing model proposed by Keim et al [90].

This approach partially follows from our work presented in [142], where several tasks and challenges related to real-time visual analytics of text streams are presented, and extends it in two directions: first, the real-time aspect is stretched to accommodate all use cases where the the users work with continuously growing data sets, but do not necessarily have the requirement to react to changes in the data as soon as they appear in the stream. Second, we assume that the data stream does not have to be a text stream.

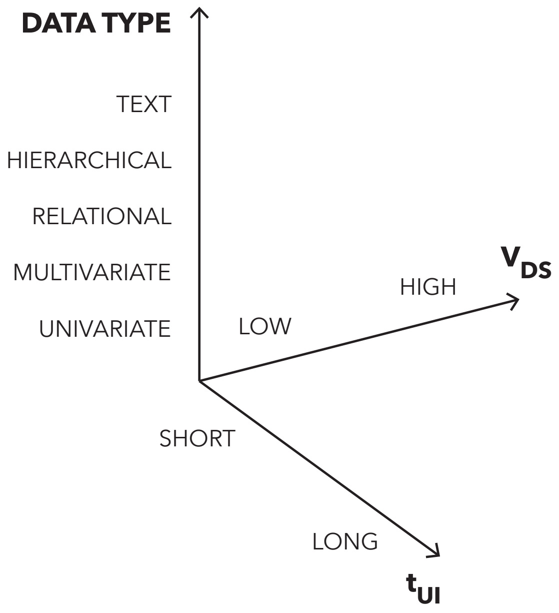


Figure 2.4: Streaming visual analytics problem space: user interaction time vs *data volume* vs *visualized data type*

Usual motivating examples for data stream analysis applications in literature involve real-time monitoring, where the analyst requires immediate response of the system in order to achieve situational awareness. Although this is a legitimate task, we believe that this requirement limits the number of applications where streaming visual analytics can play a significant role. *User interaction time* can be very *short*, such as in the previously mentioned sensor monitoring tasks, where the user is *passively* monitoring the visualization of univariate data that is being updated in real-time. However, it can also be very *long*, when data exploration requires more complex cognitive and interaction tasks on, for example, multidimensional or unstructured text data. In these cases, analysis that is being performed during t_{UI} might not be in sync with the data arrival rate and, therefore, advanced data processing and visualization updating strategies have to be taken into account.

In Figure 2.4, a sketch of the problem space in which we discuss the concept of user interaction time is shown. The data type coordinate for the space is adapted from [93]. The descriptive ranges of V_{DS} (*low* to *high*) and t_{UI} (*short* to *long*) are relative and depend on the application scenario, resource constraints and other real world requirements. We do not aim at giving a complete and precise definition of dimensions, but a conceptual overview of the reasoning space.

2.3 Challenges and Issues in Streaming Visual Analytics

This section describes a set of important high-level challenges that were derived from related work, which can be used by researchers in the future as a starting point for their work.

2.3.1 Data Handling Challenges. What to Visualize?

Basic schema in Figure 2.5 shows several important challenges that arise when dealing with streaming data, which depend on the characteristics of the data stream, its volume, existing data structures and analysis tasks.

The visualization of data streams can still be considered a broad term - we can visualize *data objects* in the stream [74] and *their relationships* [10], *pat-*

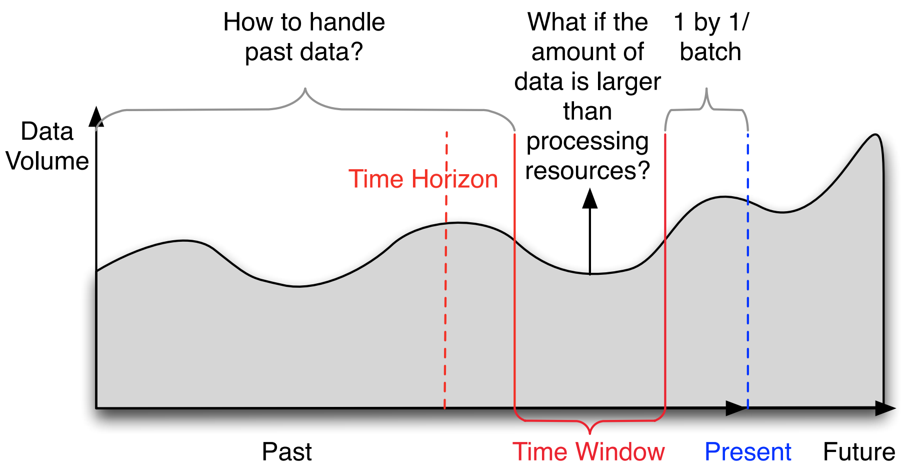


Figure 2.5: Data processing considerations for visual analysis of data streams: a) 1-by-1 vs batch processing; b) approximation of the data during bursts; c) selection of time horizon (window); d) handling of past data.

tern changes in the streams [187, 188], or *models*. After conducting interviews with analysts that need to work with dynamic data, Chin et al. [38] identified seven visual contexts that convey associations among data entities: relational, categorical, spatial, geospatial, hierarchical, temporal and overlap. Having in mind that many present data streams do not provide just univariate numerical data from a fixed number of sources, but also more complex multivariate data or unstructured text data, the mapping from the dynamic data space to the dynamic visual space becomes very challenging. In this chapter, we are referring to entities that appear in the visualization as *objects*.

How is Data Processed: Single Items, Batches and Offline

An extensive literature on data stream management covers different technical and algorithmic challenges [14]. Most importantly, fast algorithms and data structures are needed that enable real-time processing and can deal with incremental updates. Additionally, methods should not depend on apriori assumptions regarding data load, arrival rate or thresholds, because streaming data may have unpredictable contents. Moreover, we assume that streams cannot be stored completely because of their enormous size and that consequently on-the-fly processing and visualization is required. Usually, approaches either process each incoming data item individually or store data items in a buffer according to a predefined time frame, and then process the buffer content in regular intervals. However, it is not quite clear how to come up with suitable time frame sizes and consequently some approaches enable the user to modify this parameter dynamically [7, 84].

There are different ways to address some of the outlined challenges in visual analytics systems. Wong et al. [183] suggest to make the algorithmic analysis dependent on the volume of incoming data. If in the short term there is a high data load, the trade-off is between being less accurate, as in [183], and temporarily buffering data as done by Alsakran et al. [10] who buffer "document items during peak times and handle them in idling periods."

One important issue when working with text streams is performing topic modeling in real-time. Ishikawa and Hasegawa [84] cluster documents in real-time incorporating a "document forgetting model", i.e. older documents have less influence on the clusters. The clustering is an incremental k-means algorithm, which has the drawback that the parameter k has to be to predefined. Zhang et al. [196] introduce an evolutionary hierarchical Dirichlet process that allows the number of clusters to vary over time, but they do not comment on a potential real-time capability of their approach. Rose et al. [144] cluster keywords for each time interval using a hierarchical agglomerative clustering algorithm in order to learn themes without having to predefine the number of clusters. Themes of adjacent time intervals are grouped into "stories" according to their similarity. These stories can split and merge over time.

Another issue is querying in real-time. Hetzler et al. [81] allow the user to define a network combining different dynamic queries and visualize the corresponding result sets in the same structure.

2.3.2 Visualization Updating Strategies

When to Update?

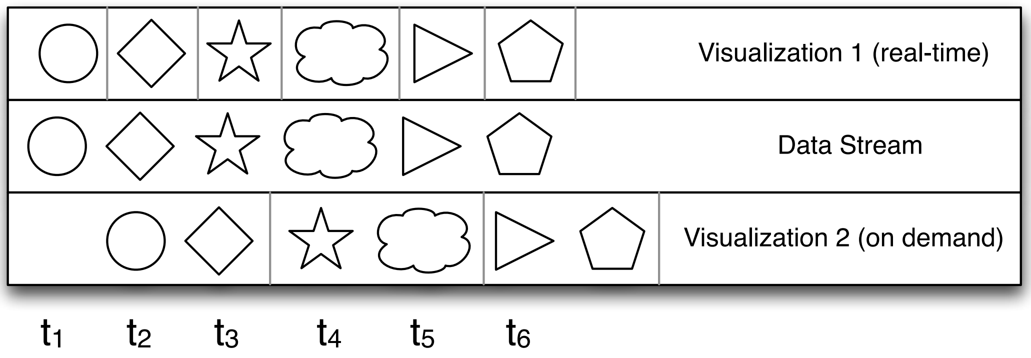


Figure 2.6: Data- and user-triggered updates of the visualization: Data objects are appearing in the data stream (middle row). In the (near) real-time visualization (top), the current state of the stream is reflected as soon as the data is processed and visualized (data-triggered update). On demand visualization provides the user-triggered snapshot of the stream.

While this question is very simple, we believe that it is actually one of the fundamental questions with far reaching consequences on the design process of a visual analytics tool. As we already described in Section 2.2.2, we have aligned the tasks on the temporal axis according to user interaction time, from quick monitoring and control (detection) tasks to slow tasks, such as exploring large text streams.

Having the complexity of user interaction tasks in mind, we can classify them into two categories: *passive* and *active*. Passive tasks are those tasks where the user interaction time t_{UI} is shorter than the time needed to process and visualize new data, t_{DS}, which means that the visualization can be updated as soon as the new data is processed. We define this mode of update as *data-triggered*. On the other hand, the active tasks are the tasks where

$t_{UI} > t_{DS}$. If the data volume is high and also contains more complex data structures, it is probably not useful to update the screen during interaction and the update of the visualization is performed on demand. This is the *user-triggered* update. Figure 2.7 aims to give a rough categorization of the update types based on the length of user tasks and processing time.

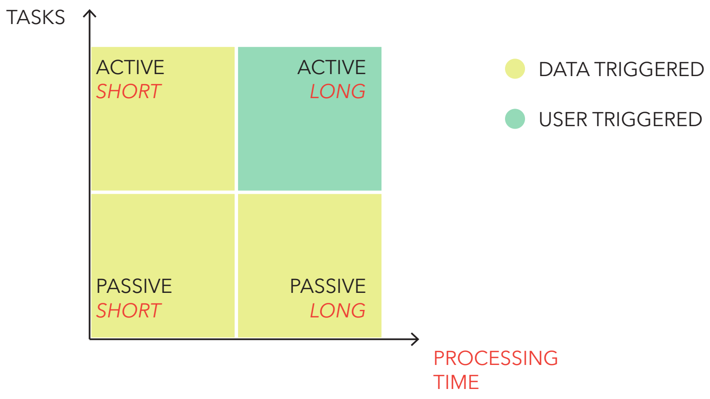


Figure 2.7: A rough categorization of user triggered and data triggered updates based on processing time and exploration tasks

In their work on concept drift classification, Zliobaite and M. Pechenizkiy [197] categorized the applications into four categories based on the type of drift and task: real-time, fixed lag, variable lag and on demand. Our data-triggered update mode can be extended and further partitioned in a similar way, where the fixed and variable lag could be more precisely measured for each of the modules in the visual analytics process. The first three categories fall under data-triggered updates, since they depend on the properties that exist in the data space, such as data stream volume, data processing module and resource characteristics.

How to Update?

As previously stated, both *data-* and *user-triggered* update types depend on the data type, volume and user, and require additional considerations in the visualization and data spaces. How is the user informed about the new data? If the volume of the data is too big to be processed in acceptable amount of time and has to be approximated [68], how do we present the approximation and uncertainty to the user? Recent work by Fisher et al. [58] presented a novel method for visualizing approximate incremental query results and uncertainty levels. The decisions made in the visualization and data spaces have to take into account e.g. layout stability, representing new data in the context of the past, rate of data change, approximation of high data volumes, etc. One of these issues is a perceptual phenomenon known as *change blindness*, when the users fail to notice large changes to visual scenes [156].

2.3.3 Encoding Data Age and Relevance

What is New?

Figure 2.8 shows an important feature of mapping from the current data streaming window to the visualization space. Usually, the mapping is not bijective, i.e. the visualization does not show only the snapshot of the latest data stream window, but also includes the data points from the past. In order to understand the current state of the data stream, the user needs to understand the change of patterns and should be able to put the new information in the context of the past.

Therefore, it is necessary to encode *the age* of displayed objects with a satisfactory level of detail.

For example, in [38], the authors enhance *treeview* and *treemap* [154] visualizations by using color to highlight the nodes in the tree structure where the new data is added. The limitation of the proposed dynamic visualizations is that the hierarchical structure is fixed and the data points can arrive only to the already existing nodes. In data streams, one of the common challenges is that the underlying hierarchy changes over time. This implies that the hierarchy has to be rebuilt in online fashion, while adapting the hierarchical visual context. In the field of graph drawing, Binucci et al. [24] proposed a method to draw trees in a streaming model. Another technique uses the color to map the age of data points [81], where the distinction be-

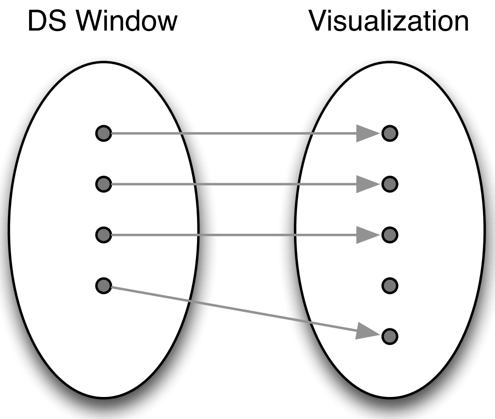


Figure 2.8: Mapping from the data streaming window to visualization is injective and non-surjective - the visualization can contain the objects that appeared in the past.

tween fresh (new) and stale (old) documents is made using a yellow color for new documents in the visualization.

This simple binary age classification can be extended to be continuous. Explicit representations of time usually use one of the axes as the timeline, along which the data points / data objects are plotted as they appear in the stream and make the object age immediately visible. Aigner et al. [5] provide a systematic overview of visualization of time-oriented data.

What is Relevant?

Adding new data points to the visualization would ultimately lead to display clutter and, in order to reduce the clutter, some data has to be removed. Most of the methods that we reviewed remove the objects from the visualization after a certain time or a certain number of new data points. This is a time-driven discarding criterium, which is often employed in the existing work (e.g. [24]). However, this is not the only approach, since very often the relevant objects can be old and they can still carry an important informative context. Keeping data objects on the screen will ultimately lead to over plotting and clutter. An extensive body of work on clutter reduction methods and techniques exist and a good overview of the topic can be found in [55].

New relevance measures need to define which data should remain on the screen, and which data should be removed. When working with a static dataset, the analyst can employ different clutter reduction techniques, such as sampling, filtering, or clustering. In a data streaming scenario, the age of objects should also be taken into account.

How to Visualize Past Data?

As previously described, removing the data from the screen reduces the clutter and can additionally improve the runtime performance of the visual analytics system. The visual objects, as depicted in Figure 2.8, can represent the data points that exist in the current time window, as well as the data points that appeared in the past. Depending on the analysis tasks and resource constraints, the objects that refer to the past data can be approximated using different aggregation methods. Xie et al. [186] used DOI (degree-of-interest) function to represent users' interest in the past data within a set of time windows. Hao et al. [74] used variable resolution density displays for visualization of univariate data, while in [73], a user-driven method for sampling time series with DOI function is presented.

The necessity to take into account the object age is related to an old body of research by DiBase et al. [48]. The authors propose three dynamic variables for designing animated maps: scene duration, rate of change between scenes, and scene order. MacEachren [118] extends this set with display date, frequency and synchronization. Although they are related to the *scene* in the animation, these variables could be a basis for streaming visualization objects variables: *birth/death*, *age*, *rate of change*, and *frequency*.

2.3.4 How to Show the Change Between Updates?

Figure 2.6 presents the concept of the data- and user-triggered visualization updates, and it also serves as an example of change representation in the streaming visualization. If the visualization is user-triggered, how do we present differences between the two updates to the user?

Considering the output of the visualization process at two consecutive time points t_i and t_{i+1}, the update can be executed either by using animation or snapshots. Albrecht-Buehler et al. [6] presented a tool for monitoring of important keywords that appear in the text stream using graph visualization. A new node in the graph appears on the outer regions of the graph and afterwards it is animated to its end position in the node-link view. Tversky and Morrison [170] discuss the efficiency of animation over static graphics and claim that "Animations are often too complex or too fast to be accurately perceived".

> There are two different approaches to visually express changes in the data. The first and more straight-forward solution is to create individual views for different time intervals and sequentially provide this series of views. Here, the challenge is to relate one view to the next view and moreover assure that the user is able to perceive the differences between both. Usually, some sort of transition is used like the animation of appearing and disappearing words in the Parallel Tag Clouds [42]. Many standard text visualizations can potentially be adapted to this scenario.
>
> The second and more demanding possibility is to provide a continuously evolving visualization reflecting a continuous data flow. In such a scenario the current view is constantly modified in accord with the newly arriving data. The modification is triggered by an update and usually expressed through motion. A nice example for the use of motion is [6]. The authors review the perception of motion and derive design guidelines for visualizing changes in live text streams. More recent approaches go in a similar direction: Dörk et al. [49] chose to "use primarily motion to represent data change" and Alsakran et al. [10] point out that "the dynamic procedure of this change is critical for reducing change blindness". However, even if the changes are traceable using motion, too many changes at once will overwhelm the user. The system of Alsakran et al. [10] addresses this issue by having at most one label change when a new document arrives and buffering documents during peak times at the cost of intro-

ducing delays.

The problem of having a big change in the display after updates also exists in visualizations that display the arrival order of items by position. Hao et al. [74] discuss different ways of shifting displayed items on the arrival of new items by minimizing the movement in the display. Chin et al. [38] discuss how standard visualizations can be extended to support the analysis of dynamic data and experiment with different ways of updating their dynamic spiral timeline, thus keeping the display more or less constant.

In the case of frequently or even continuously updating displays, support for interactive exploration is a further challenge. While a lot of approaches enable some user interactions like filtering and highlighting search queries on-the-fly, in-depth exploration is difficult. Interaction with a visualization while the stream keeps coming in, brings the problem of "adapting its display to stream content and user interaction" [6]. Those are two potentially conflicting triggers for changing the display. What if the user might like to explore a peculiarity further? Does he have to worry that it might disappear in the next moment when new data comes in (and could be lost forever)? One solution to prevent this is to freeze a certain point in time for exploration, the user can pause the stream for exploration [81, 10] or take interactive snapshots to save the current situation for a later off-line exploration as in [81]. Many other approaches, however, do not address this problem explicitly or assume that the whole stream can be saved and retrieved.

Animation or small multiples - time/space perspective

When representing the passage of time, both methods can be defined as a sequence of static images, which are ordered – either in time (animation) or space (small multiples). Since the stacking of elements of a sequence is performed in time dimension, animation provides *more space for one single image* and therefore can effectively show more data, while, on the other hand, small multiples provide more time *for understanding and analyzing what has changed*.

2.4 Incremental Visualization Properties

Based on the identified challenges, I will summarize the most important properties of incremental visualizations. They will be used in Chapter 3 for classification of existing streaming visual analytics systems.

Incremental visualization properties are closely related to *visual scalability*. Eick [54] defines visual scalability as "the capability of visualization tools to effectively display large data sets in terms of either the number or the dimension of individual data elements". Keim [91] describes scalability as a key challenge of visual analytics, as it "determines the ability to process large datasets by means of computational overhead as well as appropriate rendering techniques". Robertson et al. [139] discuss different types of scalability issues in visual analytics: *information scalability*, *visual scalability*, *display scalability*, *human scalability* and *computational scalability*.

Layout initialization

This property refers to the starting parameters of the visualization. It describes the way the visualization boundaries and initial values of the visual features are determined. It can be *static*, when the initial parameters are predefined by user, regardless of the data stream characteristics, and *dynamic*, when the initial parameters are set according to the data that appeared in the first time window.

Incrementality

Incrementality describes whether the visualization method can be effectively used with streaming data. Some methods require complete recalculation of the visualization when new data arrives, such as treemap or force-directed graph layouts, while the others allow adding new data points independently of other data, such as scatterplots. There are existing approaches, which try to overcome non-incremental nature of a specific visualization method, such as [183] for drawing MDS projections, and [24] for drawing graphs in streaming model.

Transition between the updates

The transitions between the updates can be using snapshots, small multiples or animation. The use of animation for transitions in time series has been researched in a taxonomy by Heer and Robertson [79].

Layout stability

In the context of data streams, data at time point t_i (and possibly retained data from the past) determine the visualization layout and other visualization parameters. The layout has its coordinate system in which the visual objects properties are encoded. When the new data comes in at time point t_{i+1}, the visualization should be updated to include new data values. Since the streaming data can have unbounded characteristics, scaling of the visual space is necessary when new data points have the values that exceed the boundaries of the visual features used. We can differentiate between two different classes of out-of-boundary updates:

- Update of the layout
- Update of object's visual features

The visual features can be, for example, *position*, *length*, or *color*. When mapping color to ordinal data, the minimum and maximum value for each chosen component in the color space are clearly defined. At time step t, all data values have to be mapped to the chosen color model, and they are usually normalized to show differences in the existing data. When new data has to be encoded at time step t+1, the **context** data has to acquire new values. This can lead to serious perceptual difficulties. The applications where color

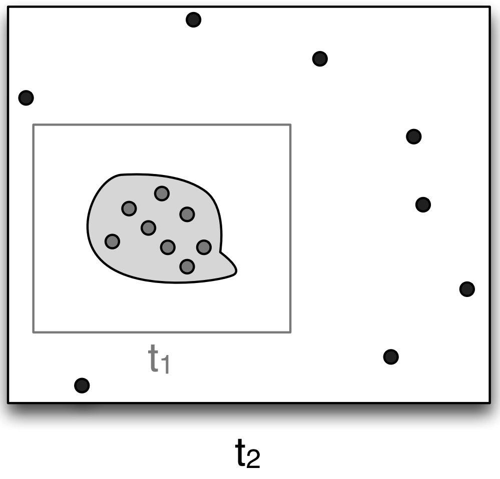


Figure 2.9: Layout stability between time points t1 and t2.

change is not an issue are those where the user needs to be aware only on the current snapshot (slice) of the data.

When reviewing the literature, I have found out that the layout stability has been rarely discussed (an exception can be found in the work by Hao et al. [74]). Due to its impact on the user it represents an important streaming visualization property and needs to be addressed by the researchers in the future. The authors in [74] devise two human evaluation criteria for measuring the usefulness of the display techniques in monitoring applications: *constancy of display* and *usage of display space*.

Object mapping is very closely related to layout stability and describes how the new data points, i.e. visual objects, are generated on the screen in relation to existing data. The mapping of a data value to an object's visual feature in relation to other data points can be: a) independent (e.g. position in scatterplots), and b) dependent (e.g. rectangle size in treemaps). When a new data point with dependent object mapping is added to a visualization, the visual features of all the other objects have to be changed to adapt to the new data.

Object lifetime

Object is the visual representation of a data point and its lifetime represents its relevance to be shown on the screen, even though it is not currently in the data stream. In some cases, the relevance is a fixed time interval (*time-driven* approaches), while sometimes it is *data-driven*.

2.5 Design Considerations

After presenting the streaming visual analytics framework, identifying challenges related to designing streaming visual analytics systems, and discussing streaming visualization properties derived from the state of the art methods, I introduce the design considerations, which should serve as a reference for future research:

1. *user interaction time*: Designing a streaming visual analytics system should include an estimation of the user interaction time required to solve the specified tasks and the data processing modules should be adapted accordingly.

2. *triggering updates*: Depending on the tasks and the expected degree of interaction, the update of the visualization can be either triggered by the changes in the data or called by the user.

3. *providing temporal context*: The user needs to understand *what* has changed since the previous update and *how* it has changed.

4. *updating strategies*: Depending on the application scenario, the streaming visualization should show either the current state of the data stream or the summarized view of the change between two updates.

5. *dynamic variables*: The following dynamic variables, which characterize streaming visualization objects, should be taken into account: *birth/death*, *age*, *rate of change*, and *frequency*.

6. *object age relevance*: Clutter reduction techniques should include the age of objects into account.

2.6 Change and Context in Information Visualization Techniques

Established methods in information visualization are typically used to visualize historical datasets. This section builds on the incremental visualization properties and challenges describes previously, and explores the degrees of freedom of several well-known visualization techniques typically used for static data visualizations. The degrees of freedom describe which independent visual variables can change in a visualization. I try to identify how these changes relate to the attribute and structural changes that can occur in a data stream. The most of the changes can lead to loss of context to the past data, from simple cases where the visualization is just minimally changed to significant changes where the whole layout has to be recomputed, each visual element moves on the screen and the previous order in each dimension is not preserved.

As described in the previous section, visualization of streaming data is strongly related to its temporal context and very often methods that map time to the horizontal axis are used to visualize the data stream. How do we define which part of past data is relevant for the current data and the current point in time? Although, the data being generated and delivered in the streams has a strong temporal component, in many cases it is not only the temporal component that the analysts are interested in. There are other important data dimensions that are equally important and time might be just an additional aspect that they care about. In those cases, we might want to rely on other visualization methods that can show other attributes better than temporal visualizations.

How should the visualization change when we add new data? Does the whole layout have to be recomputed when we add just one element like in force-directed graphs, or we can easily add it like in scatterplots? But what if the new attribute value is out of current minimum/maximum ranges? Can the user identify what is new and where did the old data go?

Even though these methods work with different data types, use different visual features and sometimes have completely orthogonal approaches to visualize data, they share some commonalities when used to displayed dynamic data. For each method, I have also discussed which changes in the data structure and data attributes (dimensions) can occur and differentiated between the *changes within range* and *changes out of range*. Our observations show that they all suffer to certain extent from the loss of context, with a degree that depends on the type of change that occurs in the data stream and few of them have issues that also exist when working with static datasets, such as overplotting and visual clutter. The summary of the results is given in the Table 2.1.

I have worked under following basic assumptions: first, the amount of new data is significantly smaller than the amount of the already processed data or it is just one new element. This way we avoid potential issues arising during the visualization initialization stages and minimize the complexity of the problem that would occur when the change between two time steps is bigger than the amount of the past data. Although some of the conclusions from our observations could also apply to big changes, I believe that it would be very difficult to describe and define such complex changes in the visualization. Therefore, I decided to concentrate on small incremental changes, which can, in some cases, already lead to significant loss of context. My second assumption is that it is important to understand the relational and temporal context between the new and old objects. Otherwise, the visualization of streaming data would be visualization of each new increment separately, which is not different from visualizing a historical dataset.

	What can change?	Loss of context
treemap	- Rectangles can change size/color/opacity - Rectangles could move in the hierarchy - Rectangles added/removed from the screen	- significant when size changes a lot - significant when rectangles move in the hierarchy - change of color (when values are out of range
scatterplot / map	- Ranges of the axes - Points added/removed from the screen - Existing points can change their properties (size, color, opacity, position)	- only when outside min/max range - minimal due to point order preservation - minimal because of the assumption: amount of change << past data - other problems: overplotting, clutter
streamgraph	- Ranges of the axes - Streams reordered - Streams added/removed from the screen	- significant due to stream reordering
line chart(s)	- Ranges of the axes - Lines added/removed from the screen	- only when outside min/max y-axis range - other significant problems: overplotting, clutter, low resolution
horizon chart(s)	- Ranges of the axes - Color ranges for new max/min values - Number of charts - Charts reordered	- change of color (when values are out of range - chart reordering
pixel-oriented (recursive pattern)	- Pixels added/removed from the screen - Color change - Dimension reordering - Recursion levels changed	- color change when values outside range - if dimension reordering needed (rare)
word cloud	- Word size, position - Words added, removed	- spiral: only for a big change - force-directed: significant due to layout recomputation

Table 2.1: Which components of different visualization techniques change when new data is added? The table lists common information visualization techniques, the visualization properties that can change in each technique and summarizes the loss of context that occurs when new data arrives in the data stream

2.6.1 Analysis of Independent Visual Variables and Loss of Context

Treemap

Treemap is a well-known space-filling technique used to visualize hierarchical data. Each node in the hierarchy is represented by a rectangle, whose area corresponds to (one of) the node's attribute(s). The fill color of the node can be used to show another data attribute. This technique can display both the hierarchy and node values without overlap and has been used extensively since its introduction in [88]. There exist several popular layout algorithms, such as squarified [28], pivot, slice-and-dice [88], spiral [168] or ordered treemap [18], which try to balance between aspect ratio, order and layout stability.

The obvious advantage of the technique to improve visibility of small items in a single layout is overshadowed by its instability when applied on streaming data and other design issues [169]. In many applications, such as monitoring stock market data, abrupt layout changes will lead to loss of relational and temporal context of the past.

Treemap belongs to the class of *relative* visualization techniques in which the features of the visual objects used to depict data items *always* depend on the features of other data items. The basic visual object in a treemap is the rectangle, whose size is always proportional to the other rectangles and the fixed size of the visualization. Typically, the first dimension from the hierarchical dataset is mapped to the *area* of the rectangles and can change as the underlying dimension in the data stream change over time.

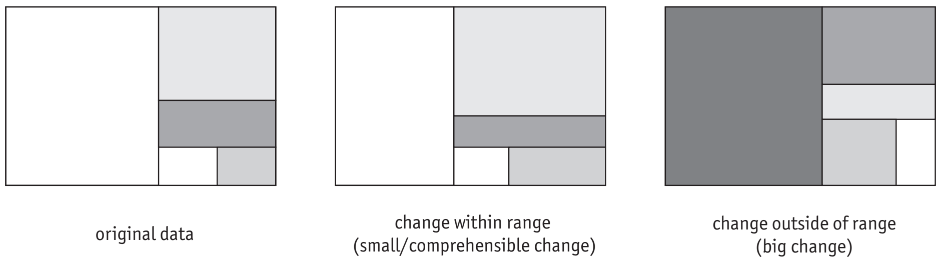


Figure 2.10: Changes in treemap. The changes in the data stream can lead to adjustments of the rectangle size, color and opacity and positioning. Although small changes might not lead to a significant loss of context (middle), bigger changes can lead to complete rearrangement of the layout, since the ranges and volume of the data stream is not predictable (right).

Next, the second dimension from the data is usually mapped to the color according to a predefined colormap. In a streaming data scenario, the data is potentially unbounded, which means that the colormap needs to be readjusted in every rectangle when only one value in the whole dataset is out of minimum/maximum range. An early attempt to map dynamic data changes in a treemap can be found in [38], where color is used to show high activity in the node. In this example, the rectangle areas do not change their fixed size and time is not mapped to any of the visualization primitives.

Treemap applied on streaming data would belong to the class of visualization techniques that provide *implicit* temporal context [142], which means that it could be used to represent new data in the context of the past data, although it would hardly be possible to reconstruct the history/evolution of the data. The color (or opacity) of the rectangle could be used to reflect recent changes by mapping the time of last change to it.

Finally, the aforementioned issues appear when the changes in the stream occur in the attribute space. Additional problems may arise when there are structural changes in the hierarchy, i.e. when the new nodes are added to the tree, the nodes are removed from the tree, or the nodes move in the hierarchy. This would lead to similar abrupt changes in the layout, which occur when the rectangles are resized.

The issues related to the changes that can occur in the streams of hierarchical data and major issues related to the loss of relational and temporal context are summarized in Figure 2.10 and the table shown in Table 2.1.

Scatterplot / Map

Scatterplot is typically used to visualize multidimensional data in order to find correlations between dimensions and detect visual clusters. Two data attributes are mapped to Cartesian coordinates, creating a point, whose visual features (size, color, shape...) can be mapped to other data attributes.

Scatterplot belongs to the class of *absolute* visualization techniques, where each visual object is placed in the visualization independently of other objects. In a data streaming scenario, if a new data item arrives, whose main attributes are within current dimension ranges, it can be placed in the visualization without losing neither the temporal nor the relational context to past data. This is shown in Figure 2.11 (middle). Minimal loss of context might occur when the data attributes are out of range, as shown in Figure 2.11 (right), which can be also applied to changes in other data attributes

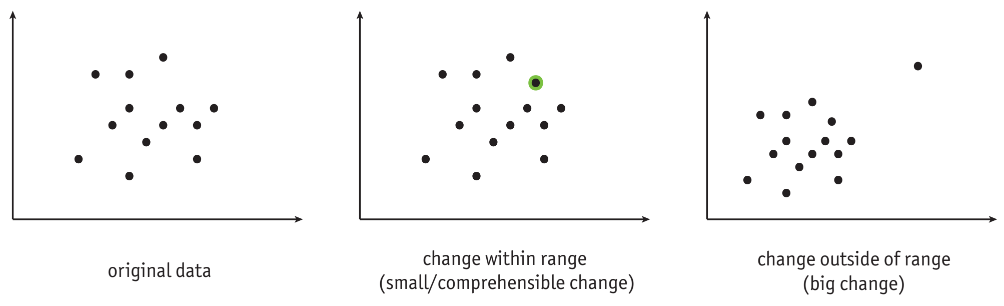


Figure 2.11: Changes in scatterplot. Adding a data point whose values are within the axes ranges (middle) causes a minimal change of the display, while adding a data point outside of the axes ranges requires the whole display to be adjusted (right).

(point size, color, opacity or shape). Although the loss of context might not necessarily be significant, this technique suffers from the problem of overplotting and clutter, which is a disadvantage that also exist when working with static dataset.

Scatterplot, similarly to treemap, belongs to the class of visualization techniques with implicit temporal context [142], and one possibility to encode time (age) of the visual elements on the screen would be to use color or opacity.

Streamgraph / ThemeRiver

Streamgraph (shown in Figure 2.12, also known as *stacked graph* or *ThemeRiver*) is a well-known technique that is used for visualizing data streams [30, 77, 49, 177]. It shows several “streams”, i.e. variables that change their values over time and are layered on top of each other and symmetrically along the timeline. This is a good example of a popular technique where the term “stream” is used incorrectly from data streaming perspective, because of its strong limitations in terms of *streamability*, which we will discuss below.

As with other visualization techniques used for time series data, the ranges of the axes can change. Since the new data is added to the right

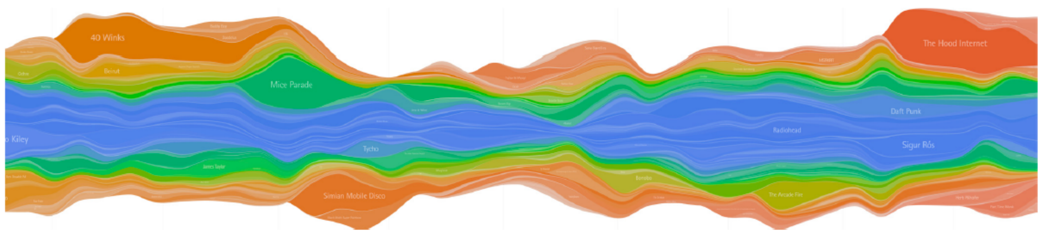


Figure 2.12: Streamgraph, also known as ThemeRiver or stacked graph. The optimal ordering of layers requires offline calculation, such that the streams with the highest "wiggles" are laid on the outside. Image taken from [30]

of the visualization, the viewport on the timeline (x-axis) can either rescale to keep all the past data, or just shift and remove the same period length on the left that has been added to the right. If the new data is out of range for the maximum values on the y-axis, the streamgraph would have to rescale vertically. These shifts and size changes do not introduce major losses of context and we treat them as negligible in our observation.

However, an important property of a "good" streamgraph is the optimal ordering of individual streams, i.e. the ordering that provides the best legibility of the visualization. The legibility criteria is discussed in [30]. In order to achieve good legibility, the visual objects (layers) with the least amount of change are positioned in the middle, while the most "bursty" layers are on the upper and lower side of the graph. This ensures minimum "wiggles" of the surrounding layers. Having a layout in which the positioning of each visual object is dependent on all the other objects, leads to the following issues when applied to the data stream: first, when new data comes in, layers have to be reordered, which requires recomputing of the whole visualization. This does not just increase the performance costs, but it can also cause confusion for the user, since the mental map of the visualization will not be preserved. Additionally, the unexpected bursts in each layer, which are characteristic for data streams, still might not result in a legible visualization even if reordering is applied. Second, if the layers are not reordered, the stability and optimal layout of the visualization is disrupted. In that case, the visualization depends on the initial layout that doesn't represent the current state of the underlying data.

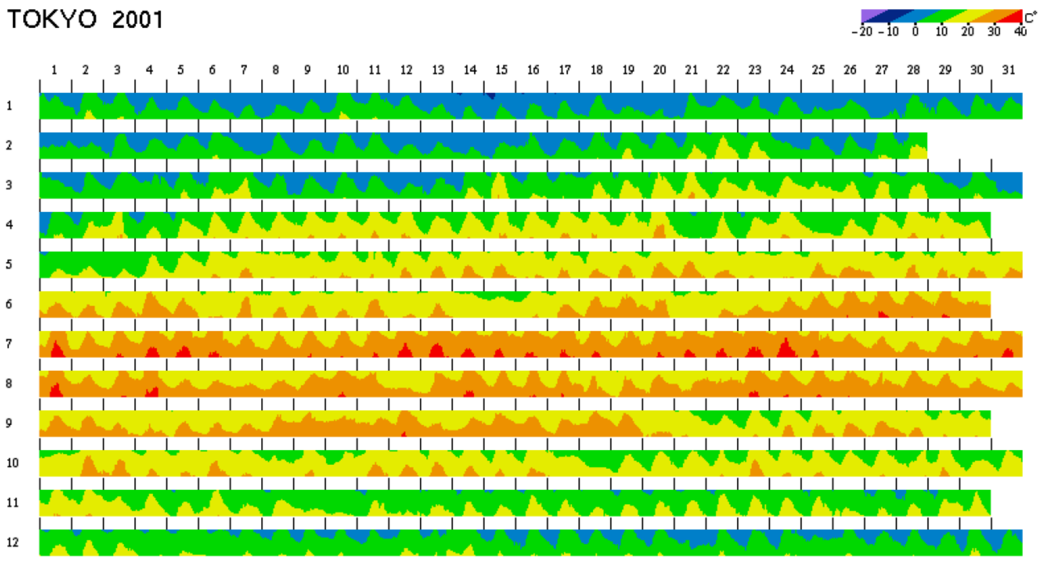


Figure 2.13: Horizon graph is very often used to visualize multiple time series. When the new data points are outside the predefined range, the colormap has to be readjusted, thus leading to the loss of context of the past data. Image taken from [147]

Having so many undesired properties and serious limitations, the streamgraph can hardly be a good choice for streaming data. However, an exception could be made if the following criteria is fulfilled:

- the bursts in each layer are not too extreme and not too frequent,
- the number of layers is low,
- the past data is discarded or approximated using, for example, multi-resolution time windows,
- the time for user exploration between two updates is relatively long,
- there is enough time to inform the user about the new reordering (for example, using animation).

Horizon graph(s)

Horizon graph 2.13, also known as ”two tone pseudo coloring” [147] is a visualization method very often recommended for displaying multiple time

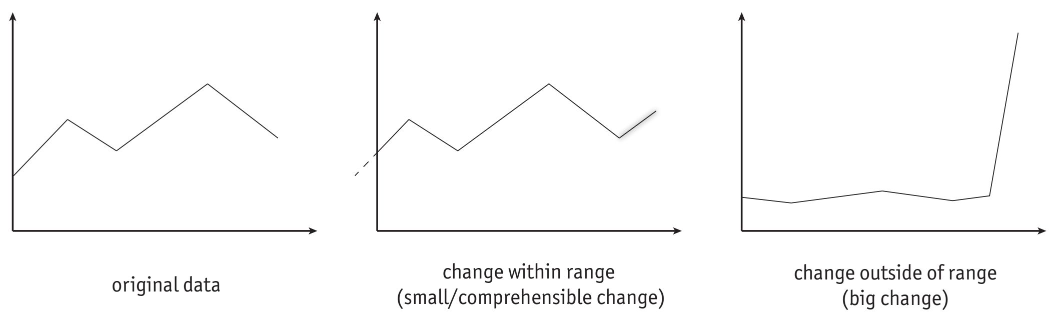


Figure 2.14: Changes in line chart. Adding new data to the current line requires readjustment of the display when the data is out of axes ranges, while adding new lines can cause overplotting and clutter.

series due to its efficient use of space and color than other time-series techniques [78, 86]. Losing the context of the past data when working with data streams can occur in two cases. First, if the new values are outside of range, the whole color map has to be reapplied to the whole visualization again. Second, the order of the horizon charts can change. Although this is not the problem of the visualization itself, it is an issue since this method is usually used to visualize multiple time series and not just one.

Line chart(s)

When applied to streaming data, this low-resolution technique has the similar properties that were described for streaming graphs - rescaling the visualization when the new data is out of range, as shown in Figure 2.14. Since it uses the explicit notion of time, the loss of temporal context would be minimal. However the well known disadvantages from static datasets, such as overplotting and clutter, exist here as well when the new variables show up and new lines have to be added to the screen.

Pixel-oriented visualizations

Pixel-based visualizations (Figure 2.15) are a set of well known techniques [92] for visualization of multidimensional data whose main idea is to use one pixel for one attribute value by assigning the value to the color of the

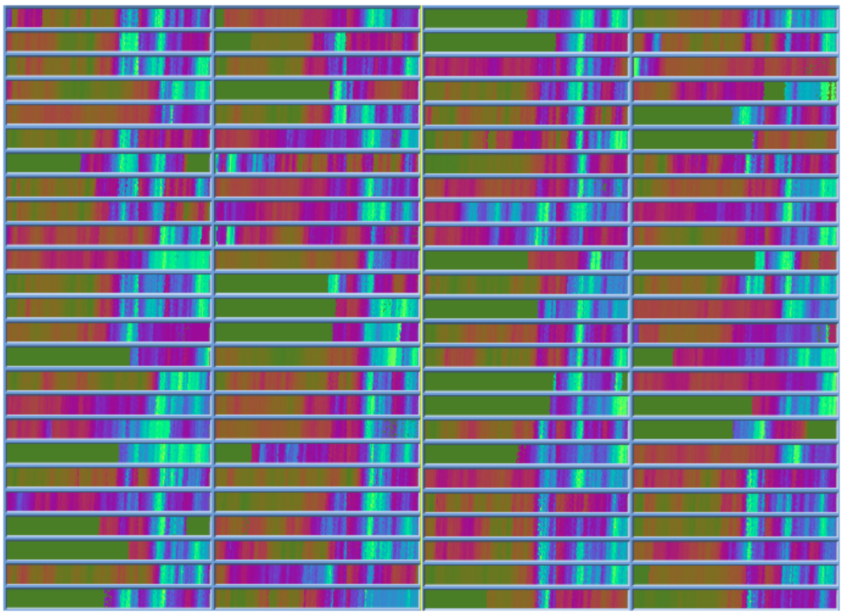


Figure 2.15: Pixel-oriented visualization: Recursive pattern technique, image taken from [94]

pixel. The pixels are arranged according to different criteria, with recursive pattern being one of the most popular arrangements [94]. The pixels are arranged according to user-defined parameters for each recursion level.

Since the data value is mapped to color, all the pixels would have to be repainted if the new data value is out of maximum range for the chosen colormap. The ordering of dimensions might be another problem that could lead to the loss of context.

Word cloud

Word cloud, shown in Figure 2.16, has been extensively used to visualize text data [89, 152, 43, 45]. The layout algorithm tries to pack the words as tightly as possible, and two approaches are commonly used: force-directed and spiral. Applying the force-directed layout to streaming text data can lead to a significance loss of context, since the new words that appear in the stream would require the whole layout to be recomputed. On the other side, spiral layout, which places the word with the highest weight in the center

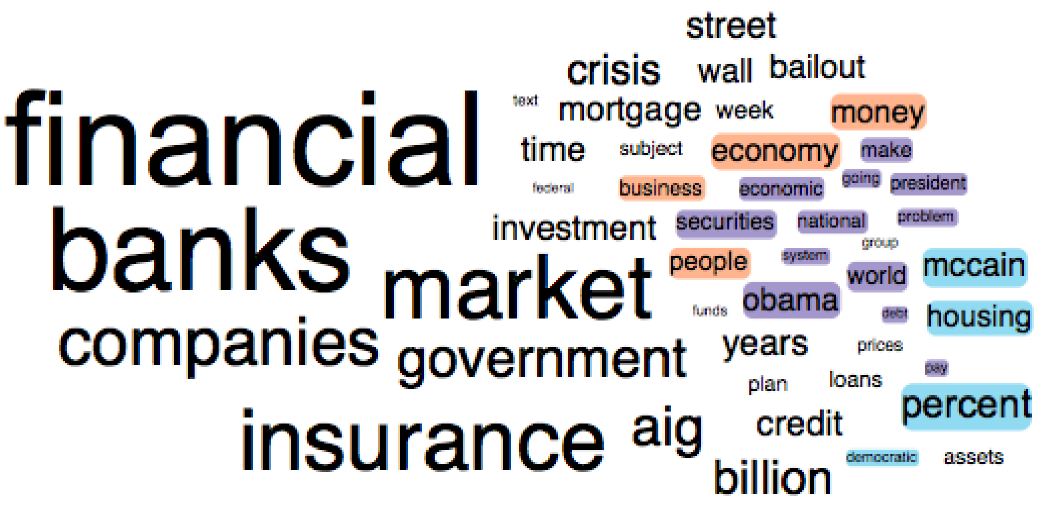


Figure 2.16: Word cloud visualization. The words are usually positioned without overlap using the force-directed method or spiral arrangement. Image taken from [45]

and then arranges the rest on a spiral path, would lead to loss of context only if the change between the updates is big.

2.6.2 Change/Context Metrics and Criteria

In the previous section we have described which properties of different well-known visualization methods can change when working with data streams. As it can be seen in the Table 2.1, the majority of problems that occur are related to the loss of context, while few methods suffer from problems that already exist with large static datasets, such as overplotting or clutter in scatterplots and line charts. Our first step in identifying and describing these problems should be extended to quantifying the loss of context and defining other optimization, stability and aesthetic criteria for streaming data visualization.

One such metric could be similar to the layout distance change function proposed by Bederson and Shneiderman [18]. This metric tries to capture how much two treemap layouts differ on average by measuring the Euclidean distances between each pair of corresponding rectangles. This metric could easily be extended by taking into account variance distance change, as proposed in [168] to detect changes when few items move by large distances while the most items do not move at all.

Hao et al. [74] propose a similar metric, *constancy of display*, which "counts the number of pixel changes per time unit" and *usage of display space*, which suggests that the empty space in the visualization should be avoided. However, they suggest that these might not be the best metrics and that other metrics should be developed. One proposal is to use the *pixel coherence*, which captures the neighbouring pixels that form a pattern and are percieved as an object. It has been suggested that measuring the data shifting to evaluate the usefulness of a visualization is strongly data dependent.

However, these metrics might not be appropriate for scatterplots and other techniques in which rescaling of axes occurs (see Figure 2.11). In such cases, the movement/change of the objects (points, lines, etc), on the screen might be significant, although it wouldn't necessary lead to the loss of context. This example shows how an abstract structural information can be easily formed when the data allows it and relates to the well known *mental map preservation* criterion.

In any case, these criteria have to be taken into account together with aesthetic criteria, which are often conflicting (as seen in the streamgraph example). Beck et al. define different aesthetic criteria for graphs in [16], which, beside mental map preservation, include reduction of cognitive load and minimization of temporal aliases that occur when a node is positioned after the update in the same place where another node was before the update. A survey on visualization of large graphs [171] reviews general, dynamic and scalability criteria for dynamic graphs and discusses two categories of visual display of time changes on graph elements: animation and static displays.

2.7 Conclusion

In this chapter, I identified three processes that need to be taken into account when designing streaming visual analytics systems: 1) data stream itself with its dynamic and complex properties; 2) data processing module, which is characterized by running times of its submodules; and 3) user interaction time, which depends on the specified tasks and defines the parameters of the other two processes. These processes provide a general concept of introducing temporal aspect into the visual analytics paradigm. Next, I have identified a set of challenges that exist in visual analytics for streaming data with a focus on the visualization space and identified the design considerations for future systems. I described the changes that can lead to loss of context in common visualization techniques when working with streaming data. The relationship between the new and old elements and temporal context that distinguishes the new and old elements is addressed. I analyzed the usage scenarios under the assumption that the data is being updated one element at the time or *delta* is much smaller than the rest of the visualization. For each technique, I differentiated between the updates that occur within the existing visualization structure and the updates with new elements in the visualization and identified possible research directions for measuring the loss of context.

3 Existing Approaches to Visual Analysis of Text Corpora

Contents

Visual analysis of events in text streams is closely related to a number of different domains in data analysis and computer science in general. This chapter provides an overview of these domains and how they relate to the work presented in this thesis. Section 3.1 describes the state of the art in visual analytics for evolving text collections.

Section 3.2 presents methods for visualization of text, which use 2D spatial arrangement of display objects to represent documents, stories, topics or large text corpora. The purpose of this section is to put the visual analysis of evolving text collections in a broader context of text visualization in general.

It also describes different tasks in analysis of large text corpora, where the temporal aspect is not the most important one.

A successful implementation of information visualization methods and visual analytics systems for text deeply relies on the choice of the underlying text mining algorithms. Section 3.3 provides an overview of the text mining methods that are related to detection, extraction and tracking of important pieces of information from text corpora, which are then used for visualization.

Section 3.4 provides an overview of the visualization methods for temporal data. The aim of this section is not to cover or classify every possible method but to identify the approaches that are most relevant for the topics discussed in this work.

Finally, Section 3.5 presents the work related to visualization of data streams in general. It follows the design considerations presented in Chapter 2, which are derived from the state of the art methods.

3.1 ThemeRiver and Beyond: Visual Exploration of Evolving Text Collections

In recent years, there has been a growing interest in the field of visual analytics for developing novel methods to analyze text collections, especially to better understand the development of topics, events, stories, themes etc. over time. These different terms relate to different models in information retrieval that are used to extract meaningful information from unstructured data, and to different concepts in journalism that are used to describe real world events.

One of the first and the most popular methods for text visualization is ThemeRiver [77]. As its name suggests, it shows popularity of different themes over time using a river metaphor, where the rivers flow along a horizontal timeline. The width of the river represents the popularity of a theme during the corresponding interval. *Themes* are basically important terms that were extracted from a text collection and popularity is in this case the number of occurrences of a term for a specific time interval. The intervals are equally distributed over time and smooth river shapes are created using spline interpolation between the intervals. The aesthetic properties and optimal layout algorithm of ThemeRiver, also known as *streamgraph* as a special form of a stacked graph, have been also discussed [30]. This highly influential visualization inspired several derivative methods, which aim to capture other properties of the explored data and allow the analysts to answer more complex questions. They usually combine different visualization techniques and different text processing techniques in order to allow more insight from the data. One important example is TIARA [177], a visual analytics system for analysis of text, which shows development of topics in collections of emails and patient records by integrating *tag clouds* in the ThemeRiver-like topic layers. It is one of the first approaches to combine topic models with interactive visualization to facilitate analysis of evolving large text collections.

One of the disadvantages of ThemeRiver is the vertical overlaying of the streams, which, although it helps in understanding the overall trends, hinders comparison of values between streams when the value changes a lot. EventRiver [117] is a visual analytics system for analysis of temporal development of news events and event groups retrieved from broadcast news data. It contributes a novel visualization technique, where bubble-like shapes are used to represent important events in the data. Rose et al. [145] show evolution of news stories over time. Fisher et al. [57] presented a keyword tracking tool that calculates the co-occurrences of most important terms in blogs. Thread Arcs [99] visualizes threads from a collection of emails. Examples of other visualizations inspired by ThemeRiver can also be found in [50, 44, 47, 49].

The number of topics that can be shown with ThemeRiver is, however, limited. One of the approaches that try to overcome this limitation is HierarchicalTopics [52], which builds meaningful hierarchies of topics using a novel Topic Rose Tree algorithm. This visual analytics system addresses three challenges related to the exploration of topics: 1) meaningful organisation of similar topics into topic groups to facilitate the navigation and analysis in the topic space; 2) inclusion of the user into the topic modelling loop by allowing user interaction; and 3) extending existing visualization methods to incorporate hierarchies and increase visual scalability.

LensRiver [66] extends the river metaphor from ThemeRiver into an analytical system for temporal analysis of unstructured text retrieved from video broadcast news. It deals with evolution of themes over time, their hierarchical structure, and employs different visual analytics techniques to perform the analysis.

3.2 Text Summarization using Spatial Based Visualization Methods

One of the popular methods for visualizing text are *tag clouds*, also known as *word clouds*. A tag cloud summarizes a document by arranging its most important words in two-dimensional space and assigning the font size for each word according to its frequency in the text. The aim of the arrangement algorithm is to pack the words as tightly as possible and common layouts include spiral and force-directed methods. The advantages of tag clouds is that they are simple but efficiently summarize the document and help in forming a general impression [138]. However, they usually rely on the bag-of-words model, which completely erases the document structure, semantics and word order, and they also suffer from different perceptual issues related to spatial arrangement [138]. Another important disadvantage is their dependency on the term extraction method - they usually rely on simple term counts or variants of tf-idf weights [150], although these might not be the most descriptive terms. Parallel Tag Clouds [42] combines graphical elements of parallel coordinates with tag clouds to display changes across other data attributes such as time.

An important text visualization initiative was Spatial Paradigm for Information Retrieval and Exploration (SPIRE) Project [181], whose goal was to explore the possibility of finding means to visualize text in order to reduce information processing load and to improve productivity for intelligence analysis [182]. The main visualizations were Galaxies and ThemeScape, two MDS-based projections of high dimensional document vectors. Newsmap [179], which uses news aggregated by Google, shows the data visually encoded into a hierarchical Treemap visualization, based on the amount of news in each cluster and the corresponding category.

3.3 Visual Analytics and Interpretation of Topic Models

In visual analytics, the user can visualize either the raw data, the model, or both, transform and filter the data or refine the parameters of the model during the analysis [166]. Research on visual analysis of topic development and analysis of temporal dynamics in information streams heavily relies on the underlying topic modeling algorithms and their parameters. The idea behind topic modeling is to identify persisting sets of words in large text corpora and reduce the overwhelming amount of text to a more comprehensible level for the analyst. The models are abstract structures that mold the raw data in a way that allows the analyst to interpret it according to his hypotheses and gain new insights.

Temporal analysis of news is not just a question of visual representation of events over time, but also a fundamental problem in textual data mining. An issue of considerable interest is analysis of news articles as document streams that arrive continuously over time. Each stream is not only an independent sequence of documents, but it also exhibits braided and episodic character [103]. Moreover, in today's news reporting, the most attention is paid to breaking news about the latest events, which are characterized by fast growth of amount of information until a certain peak is reached, and fading of interest afterwards. A formal approach to model *burst of activity* of topics appearing as document streams is presented in [102]. Furthermore, the propagation of short quotes over news websites and blogs is analyzed in [113].

A well-known initiative in text mining was Topic Detection and Tracking

(TDT) [8], which investigated methods of discovering events in broadcast news streams. One of the frequently used methods for extracting topics from document corpora is Latent Dirichlet allocation (LDA) [27], which identifies latent topics by learning distributions of co-occurring words and assumes that each document can be represented as a mixture of topics. Variants of LDA extend it on dynamic topic models to analyze evolution over time [26] [174], hierarchical topics [25], connect authors and topics [161], and labels [135].

However, the quality of topics extracted by these methods can be questionable when applied on real world datasets and usually have to be manually validated by domain experts. Recently, Chuang et al. proposed a framework to support a large-scale assessment of topical relevance [39]. Another approach to deal with this challenge is to take into account novel design considerations *interpretation* and *trust*, as proposed by Chuang et al. [41].

3.4 Visualization of Time Series

An extensive body of research has been conducted on visual analysis and visualization of time series. A categorization of different approaches is described by Aigner et al. [5]. In [125], Müller and Schumann provide a review of time-dependent data techniques. TimeSearcher Project [1]generated several publications on dynamic querying of time-oriented data. Pixel visualization for time series in circle segments is described in [12], time series bitmaps used for working with large time-series databases in [111], while more details about multi-resolution techniques for large time series can be found in [73]. Javed et al. [87] compare different techniques for visualization of multiple time series. In [131], Palpanas et al. describe amnesic functions for approximation of the time series data with fidelity proportional to its age.

SignalLens [101] is a method for Focus+Context analysis of single long time series from electronic measurements. Details about lens construction can be found in the Framework for unifying presentation space by Carpen-

[1]http://www.cs.umd.edu/hcil/timesearcher/

dale and Montagnese [32] and details about achieving high magnification in [33]. Sigma lenses [132] use dynamic translucence to create transition between focus and context.

3.5 Visualization for Data Streams

The chapter is concluded with a systematization of existing approaches for visual analysis of data streams (Table 3.1), based on the visualization design principles presented in Chapter 2. Data streams are continuously updated, and time-stamped data arrive in rapid, time-varying fashion making the volume of the stream unpredictable and unbounded [14]. Design of effective visualizations is strongly constrained by these dynamic properties [38]. LiveRac, a tool for visual exploration of system management time series data is described in [122], where semantic zooming of time series is presented. Although the challenges were described several years ago [183], a lot of work in the field of streaming data and text visualization still has to be done.

An algorithm for real-time news aggregation into article threads with a dynamic visualization technique is presented in [105], where important topics are shown aligned on a time axis to give insight into their temporal development. Incremental change in collection of documents, which are projected in 2D plane by highlighting new (fresh) and old (stale) documents is presented in [81].

The classification criteria are described in detail in Chapter 2.2. During the literature review process, I have considered the related work in the databases, data mining, and visualization communities. The research on data streams has gained significantly more attention in the areas of data management and data mining than in the visualization community and, therefore, the generated output is considerably larger and deals with specific challenges that arise in those domains. There exist a number of publications that cover the challenges in resource management, data synopsis, approximation algorithms, mining algorithms and concept drift. My focus is on the existing analytical systems that have the user in the interaction loop.

	Layout initialization	**Visualization incrementality**	**Transition between the updates**	**Layout stability**	**Object lifetime ("trailing")**
VADDS: Dynamic TreeView/Treemap [12]a	fixed	non-incremental	animation	dependent	time-driven
VADDS: Dynamic Spiral Timeline [12]b	fixed	non-incremental	animation	independent	time-driven
Multi-Resolution Techniques for Visual Exploration of Large Time-Series Data [21]	fixed	incremental (pixel timeline)	n/a	independent	time-driven or data-driven
Dynamic Visualization of Transient Data Streams [45]	yes	incremental (MDS)	snapshots	independent (MDS)	none
EventRiver [32]	yes	non-incremental	n/a	dependent	time-driven
BinX- Dynamic Exploration of Time Series Datasets Across Aggregation Levels [6]	no	incremental (line chart)	animation	independent	time-driven
Density Displays For Data Stream Monitoring [22]	fixed	non-incremental	n/a	independent	time-driven
STREAMIT: Dynamic Visualization and Interactive Exploration of Text Streams [4]	rescaling	incremental	animation	dependent (force-directed)	time-driven
TextPool: Visualizing Live Text Streams [3]	yes	non-incremental (graph)	animation	dependent (force-directed)	time-driven
Drawing Trees in a Streaming Model [9]	yes	incremental (graph)	n/a	independent	data-driven
Turning the Bucket of Text into a Pipe [26]	yes	non-incremental	snapshots	independent	time-driven
Visual Analysis of Multivariate Data Streams Based on DOI Functions [46]	yes	incremental	snapshots	independent (scatterplot)	time-driven
Visual Exploration of Stream Pattern Changes Using a Data-Driven Framework [47]	yes	incremental	snapshots	independent	time-driven
CloudLines: Compact Display of Event Episodes in Multiple Time-Series [30]	fixed	incremental	animation	independent	data-driven
Incremental Visual Text Analytics of News Story Development [31]	fixed	incremental	none	dependent	time-driven

Figure 3.1: Classification of streaming visual analytics papers by streaming visualization properties.

The research in the field of visual analytics has been focusing sporadically on data streaming applications, without a clear systematic overview of the challenges that arise when the visualization supports transient data streams and not the persistent relations that exist in the databases. I have also surveyed the methods that are relevant to streaming visual analytics because of the significant topic overlap, such as research papers on visualization of temporal data, role of animation, and dynamic visualization, but also the papers that deal with issues of general importance, such as visual scalability and mental map preservation.

While I aimed at providing an objective overview following a systematic approach, the heterogeneity of the approaches in the papers and the wide scope of the problem domain showed that a tighter integration of knowledge from different areas is needed in future research. I believe that my classification of the related work in visual analytics of streaming data, the issues, their implications and the guidelines should serve as a good starting point for researchers interested in this domain.

4
Interactive Exploration of Event Episodes in News Streams

Contents

This chapter introduces a visual analytics method for detection and analysis of event episodes in multiple event sequences. The method, called *CloudLines*, provides an overview of sequences while allowing direct interaction with each event in limited space. Although it has been developed with news articles in mind, it can be easily applied to any event data arriving at irregular intervals when the event type is known in advance. The basic data flowchart is given upfront as a reminder of the two approaches presented in this thesis (Figure 4.1). The fundamental assumption is that the event type is known in advance.

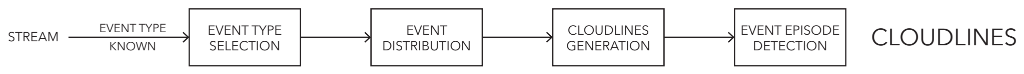


Figure 4.1: Basic CloudLines pipeline: Visualization of events whose type is known in advance.

In this chapter, the following contributions will be presented:

- A compact visualization for event sequences, which: 1) takes into account the distribution of the data points, 2) shows recent data with minimal overlap, and 3) allows easy detection of important event episodes.
- A lens-based interaction for direct access to overlapping events.
- Demonstration of the effectiveness of the approach by applying it to multiple temporal event sequences acquired from the real-world news streams.

The chapter is based on the following publication:

Milos Krstajic, Enrico Bertini and Daniel Keim. CloudLines: Compact Display of Event Episodes in Multiple Time-Series. IEEE Transactions on Visualization and Computer Graphics, 17:2432-2439, 2011. [104][1]

4.1 Introduction

Existing techniques for visualization of text collections support many analytical tasks, from providing an overview of a single document to analyzing different text features in large text corpora. Typically, they can show the development of different topics in the corpora using a ThemeRiver-like metaphor [77] by aggregating the documents in the data space over predefined time intervals. They provide a nice overview of the dataset although exhibit several disadvantages when it comes to comparison of values across different layers [30]. What is common among these visualization methods is that they aggregate the *events* (news articles, twitter messages, etc) in predefined time intervals and require an additional visual transformation step to get from the aggregate to the document itself. Furthermore, temporal data is often very long, making it difficult to interact with the individual events.

Another feature of the existing methods is that they usually treat the present and the past visually equally. In many analytical tasks, however, the user already has some knowledge about the past and wants to be informed about what is new. He also needs to be able to put the current information in the context of the past.

But how can we visualize each event individually and maintain the overview of the whole dataset at the same time? How can we still perceive temporal trends that occur when an interesting series of events occurs? Can we effectively compare several event sequences at once?

[1] I have written the most of the publication myself and Enrico Bertini helped in structuring the paper, improving the writing and proofreading. The technique was iteratively designed and developed in joint research sessions. Daniel Keim and Enrico Bertini gave advice and direction during the development of the technique and the paper writing process.

In order to deal with these challenges, I have developed *CloudLines*, a novel interactive visualization method suited for analysis of event episodes in several event sequences at once. It provides an overview of the dataset and allows direct interaction with single events.

4.2 Problem Description

A typical example where all of these questions and challenges occur is analysis of news data. Online news sources publish news that report on real-world events of different importance[2]. The data is generated at non-equidistant time intervals, making it different from other time series data, such as any type of measurement data, which is sampled at fixed time intervals. This means that each data item in the news stream carries valuable information in both its timestamp and its content. A short time between several events of the same type that appeared in the stream could mean that something important might be going on. In the world of news, these events are articles that are being published by different sources but are related to the same real-world event. They create *event episodes* that differ in importance based on their volume and time density. This dual nature of such event-based data creates significant challenges for its analysis.

My method tackles the problem of designing a compact visual representation for multiple long event sequences, which provides both a global overview of the whole dataset and allows direct interaction with single events. Besides the fact that the limited screen real estate causes high overlapping of these data, we are also addressing the problem of putting the more relevant recent data in the context of the past. This space-efficient solution helps in comparing multiple temporal sequences at once, while it can also be used for displaying single time series when vertical space is limited (for example, on the screens of mobile devices). Different timeline distortion techniques are developed to accommodate more space and allow direct interaction with the

[2]Please note the difference between the real-world event and our definition of an event. Several events in our data can report on the same real-world event, thus creating an event episode.

most recent events. At the same time, we employ Focus+Context approach to allow inspection of events in highly compressed areas.

4.3 CloudLines Design: Basic Features

Each event is displayed on a linear timeline as a circle with a minimal radius that allows interaction. Circular objects allow better distinction than rectangular in case of slight overlap. The circles are resized relatively to the temporal closeness of other events - if there are many events around an event, the circle becomes proportionally bigger. If the recent events have higher importance for the user, the timeline can be logarithmically distorted to provide easy access to these events while keeping an informative context of the past. This approach allows for smooth transition within the distorted display space, while other approaches that deal with space partitioning were proposed in the past [73].

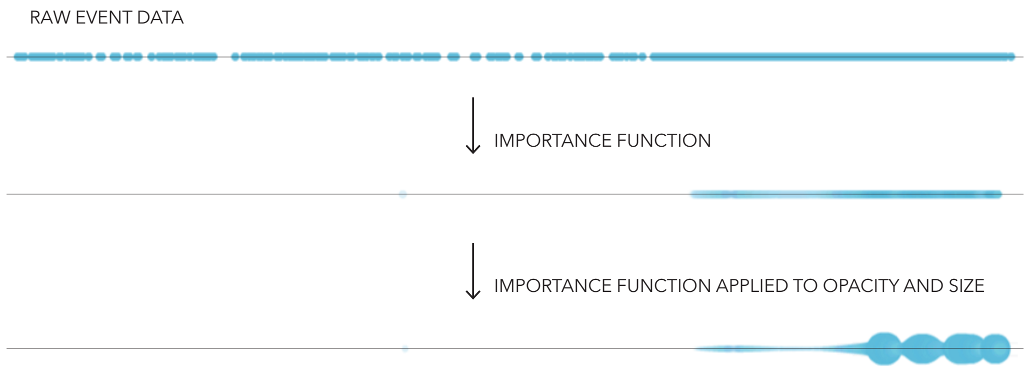


Figure 4.2: Step-by-step construction of a single CloudLine. (top) An event sequence shown on a linear timeline, where each event is represented by a circle. It is practically impossible to distinguish between the high and low density areas. (middle) The same sequence after mapping the importance function to opacity of each circle. Importance function is described in Section 4.3. The change in opacity levels shows the structure of the sequence better. However, overplotting and the fact that maximum opacity can be reached very quickly don't allow for finer details in high density areas to be shown. (bottom) Importance function mapped to both opacity and size of the circles. Several event episodes can be identified much more easily.

4.3.1 How Many Events Can We Fit on the Screen When a Sequence is Very Long?

Let's take a naive approach. Linear scaling of one month of event data without overlapping and aggregation on a 1,200 pixels wide display with the event object width of 5 pixels would allow a total of 240 events. By allowing an overlap of 80%, this number could go up to 1,200, thus creating a line of indistinguishable event data. By using object transparency the theoretical upper limit would be 120,000 events (assuming that each object is shown with 1% opacity). However, this hypothetical example would make individual item inspection completely impossible. In practice, real data have different distributions with high density intervals during important event episodes and low density areas otherwise. Fitting one month of news events on one timeline allows only 40 pixels per day, which is less than 2 pixels per hour (Figure 4.3).

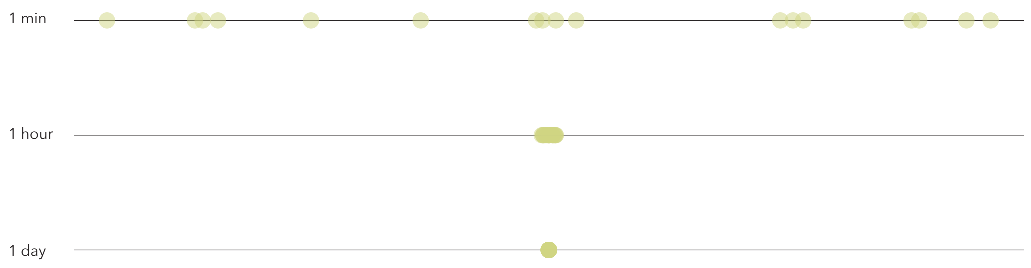
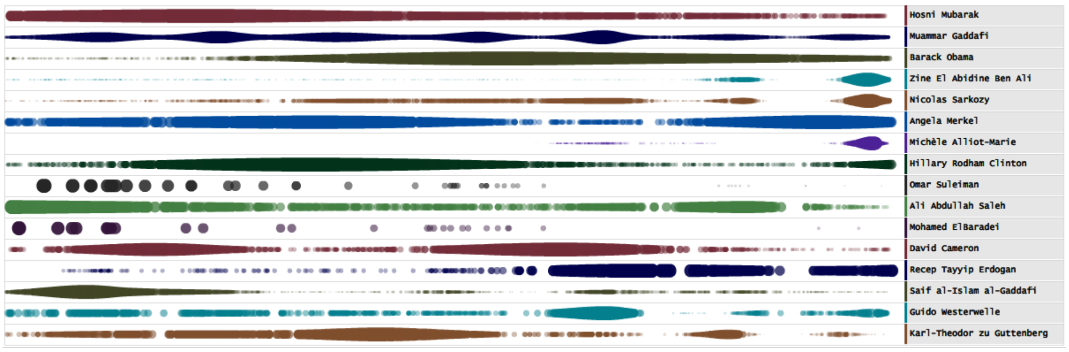
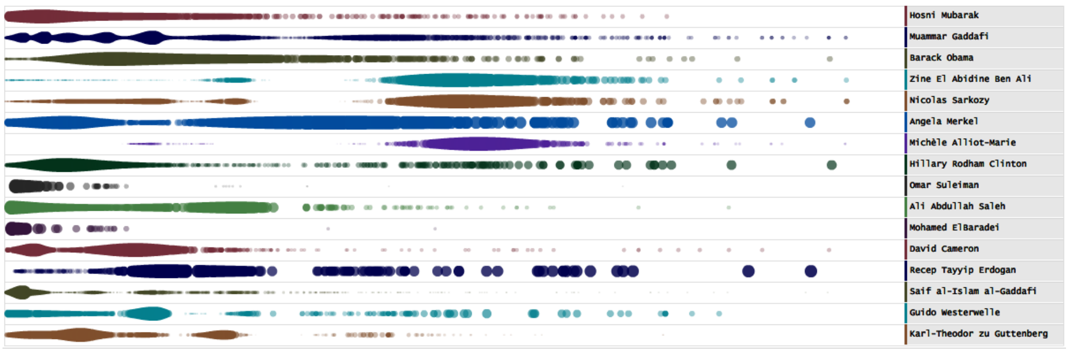


Figure 4.3: Overlap of events in long event sequences. 17 events irregularly distributed within one minute (top). The same events in one hour window (middle) and one day window (bottom). These events in a long sequence are not visible anymore.

By giving more space for recent data, and compressing the past, under the same circumstances, we provide much more space for the inspection of individual data items in the present, while creating high compression of the data in the past.



(a) CloudLines for 16 world political leaders collected from news articles during one week. Events are placed on the timeline with a linear scale. The data used in this figure is described in detail in Section 4.5.



(b) The same data aligned on the timeline with a logarithmic scale. Although both visualizations allow detection of event episodes, logarithmic timeline allows direct interaction with recent events. The past events are compressing on the left side and provide an informative context for the analyst.

Figure 4.4: CloudLines with linear and logarithmic timelines

4.3.2 Event Episodes as Visual Clusters

In a less radical scenario, where the ratio of *available space* to *observed time window of interest* is more favourable and the distortion of the time axis is not needed, our visualization can allow the user to easily distinguish between areas with high density (event episodes) and areas with low density. Typically, detection of event episodes can be solved by combining data aggregation and different visualization techniques, e.g. by plotting frequencies at discrete time points, interpolation and stacking [77] or histogram and cubic spline interpolation [117]. However, these approaches display precalculated clusters of data, while we are working on the atomic level, which reveals finer structure of the data and shifts the decision making process of what is a cluster from an automated algorithm to the user. Also, this approach makes the precomputation less expensive.

4.3.3 Use of Color

Multiple time series require compact visualization technique that will use as little amount of vertical space as possible. To support easy row identification, besides using positioning, we are coloring time series converted from color map suggested by ColorBrewer [76], which suggests a total of 11 different colors for qualitative data. Since the positioning along the y-axis is fixed in our case, we can repeat this color map on a larger number of categories without confusing the user. During our experiments with different time frames, datasets, and distortion functions, we noticed that the technique also works well in analysis of single time-series data for getting a quick overview and detection of potentially interesting areas. In these cases, color could be mapped to a different feature of the data.

By using real-world data, we have identified two overlapping factors that make detection of event episodes difficult, regardless of the linear or distorted timeline. The first factor is overlapping that is due to limited space, which can lead to event episodes overlapping each other. This effect is especially noticeable when working with distorted timelines. The second factor is the low density event data that is actually noise. To deal with this problem, we combine *kernel density estimation*, *truncation function* and *lens distortion as focus plus context technique*.

4.3.4 Mapping Event Relevance Using Kernel Density Estimation

The overplotting effects in time series are a well known problem that may be solved to some extent by aggregation or sampling, thus leading to the loss of information [73]. In our approach, we are keeping aggregation to the minimum and leave the visual cluster identification to the user. Since we have a high overlap of event data in the past due to high logarithmic compression, we need to find a way to penalize low density regions and reward high density regions.

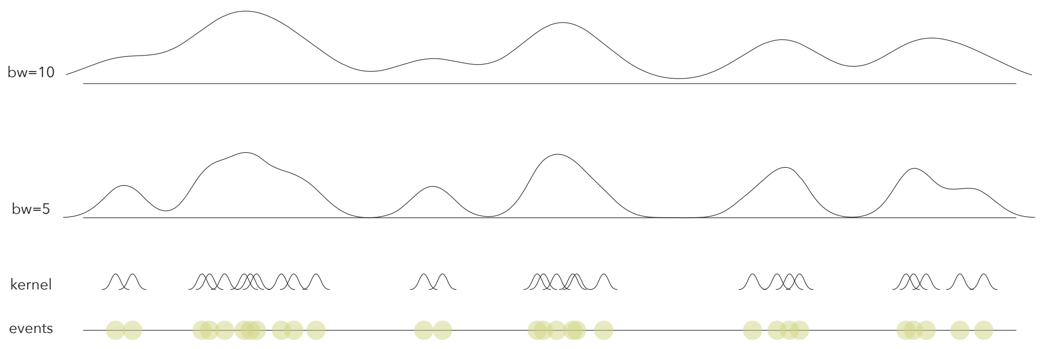


Figure 4.5: Kernel density curves of event data with Gaussian kernels and two different smoothing factors. By default, the bandwidth is estimated using Silverman's rule of thumb [155] and can be changed by the user.

As we have described earlier, overplotting can be caused by different factors and can not be easily avoided when working with data on atomic level. To overcome this problem, we propose a method that combines event importance function with density estimation and cut-off (truncation) function. Kernel density estimation is a non-parametric method for estimation of probability density function of a variable. The extensive theoretical background can be found in [173] and a simple example of constructing a kernel density estimate from several data points using Gaussian kernel is shown in Figure 4.5. The idea behind using kernel estimator is to amplify the signal coming from the high dense areas and at the same time to reduce the signal in the low density areas.

By default, the bandwidth of the kernel is estimated using Silverman's rule of thumb [155]. The effect of different kernel bandwidths on density estimation curve and comparison with histogram and logspline interpolation are shown in Figure 4.6. The user can change the parameters in order to identify global and local trends in the data.

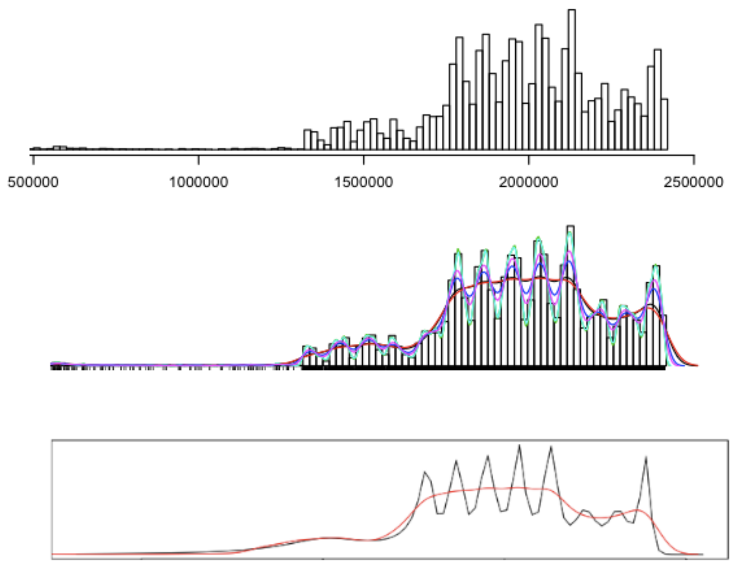


Figure 4.6: Different density estimation methods: (top) histogram; (middle) kernel density estimation with different smoothing parameters (bandwidths) (bottom) kernel density estimation with Gaussian bandwidth and logspline interpolation

4.3.5 Importance Function

For an event e_i, where $i \in (1, ..., N)$, importance function $imp(e_i)$ is defined as

$$imp(e_i) = f(t) \times g(d_i(t)) \tag{4.1}$$

where t is the *age* of the item e_i, $f(t)$ is *decay function* and $g(d_i(t))$ is a *density factor*.

Importance function for each individual item is mapped to its size and opacity. Decay function is position dependent part of the importance function. Depending on the amount of compression, noise and data volume, $f(t)$ can be defined as exponential decay function:

$$f(t) = f(0) \times e^{(-t/\tau)} \tag{4.2}$$

where τ is mean lifetime and f(0) = 1.

Density factor is density-dependent part of the importance function and it is calculated as

$$g(d_i(t)) = Y_{max} \times \frac{d_i(t)}{D_{max}(t)} \tag{4.3}$$

Here, $d_i(t)$ is kernel density estimate of event e_i at time t, $D_{max}(t)$ is the maximum density estimate in the current time window, and Y_{max} is coefficient used to normalize the height of the objects to maximum allowed height for each time series. We use the density factor for highly compressed areas, where the overlap is high because of the size of each individual item, thus rewarding high density areas and penalizing low density areas. It is recalculated at user-specified time intervals to accommodate new data, with smooth transitions between the old and the new values.

4.3.6 Cut-Off Function for Noise Reduction

By estimating densities of our event data points, we decreased the impact of low density areas on the information display. However, the high volume of long time-series in these areas can still significantly affect the calculations, interaction and rendering of the visualization. This is especially the case when long time series contain *bursts* [102] of events around few peaks. When working with news data, like in our motivating example, we can expect to have such *data shapes* and low density areas can be treated as noise. Time-

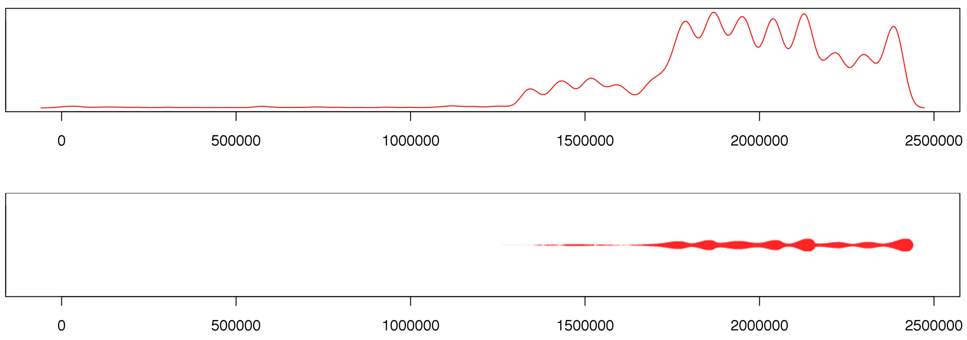


Figure 4.7: Linechart vs CloudLine. Compact design of CloudLine saves expensive screen real estate, while keeping important visual features of the time series shape.

series can be truncated to areas around the peak, like in [193], where data is truncated from 1 year to 128 hours, when it falls to around 10% of the peak value.

In our case, the newer data is considered more important than the older data and this method can't be directly applied. Since the old events are less important then the new events, the noise removal threshold should increase with the age of data. Therefore, a cut-off function is needed, which is used to decide if the item e$_i$ should be visible or not. As a positive side effect, the compressed old data will leave only the most important event episodes due to a stronger truncation.

CloudLines technique implements different cut-off functions, such as logarithmic, square root, exponential or quadratic to compare their effect. Experiments with different datasets showed that concave functions in general give better results.

$$cutoff(x) = \frac{D_{max}(t)}{X_{max}} \times log(\frac{X_{max}}{T_{max}} \times t) \tag{4.4}$$

where $D_{max}(t)$ is the maximum density in the current time window, X_{max} is the width of the display, and T_{max} is the width of the time window. Therefore, if density value d_i of an event e_i happening at time t_i is greater than the value of the cut-off function at time t_i, the item should be visible.

The results of mapping the importance function and the cut-off function to opacity and size of objects representing event data in a single time-series are shown in Figure 4.2. To demonstrate the idea of saving screen space with our technique, we have compared a line chart and a CloudLine, showing the same sample dataset, in Figure 4.7.

Another interesting feature of the CloudLines visualization technique is its applicability to inline text insertions: . This can be an alternative to the popular Sparklines visualization.

4.3.7 Static vs Dynamic Visualization

The technique could be used in both static and dynamic scenarios. It is known that animation is not suitable for describing change of many objects on the screen, but under some circumstances, when the user is given the possibility to control the playback, the animation can be useful. In these cases, it is necessary to find a good way to show new data that was acquired while the user was analyzing the old dataset.

In our tool, it is possible to automatically adjust the time window of the observed time series and switch between the static visualization, where new data is added at fixed display coordinates, and the dynamic visualization, where data items move from present to the past according to the underlying trajectory function. In our approach, although not all of the used techniques and algorithms are incremental by nature, it would be possible to extend the approach into a fully incremental solution. Currently, the algorithm used to calculate kernel density estimates at observed data points is rerun at constant user-defined time intervals on the whole dataset, but it could be replaced with an online kernel density estimation algorithm, to adapt the densities on the incremental updates only. To avoid abrupt changes in the display that could be caused by sudden change of the density values, we used smooth animated transitions that allow the user to understand the changes.

4.4 Event Interaction Techniques

4.4.1 Lens Magnification with Event Data

The visual density of the display can lead to the occlusion of fine details in a large dataset. This is especially noticeable in the regions of ”maximum age” and high data density, when the logarithmic distortion of the timeline is used. To overcome this problem, we couple proposed visualization method with magnification lens distortion technique, which allows a fine-detailed analysis of individual event data points. Focus+Context approach provides a detailed view [62] of an interesting short interval within the global overview of the long time series. The proposed focus plus context technique is further improved to allow undistorted view in the focus region, while providing smooth transition in the areas connecting the focus

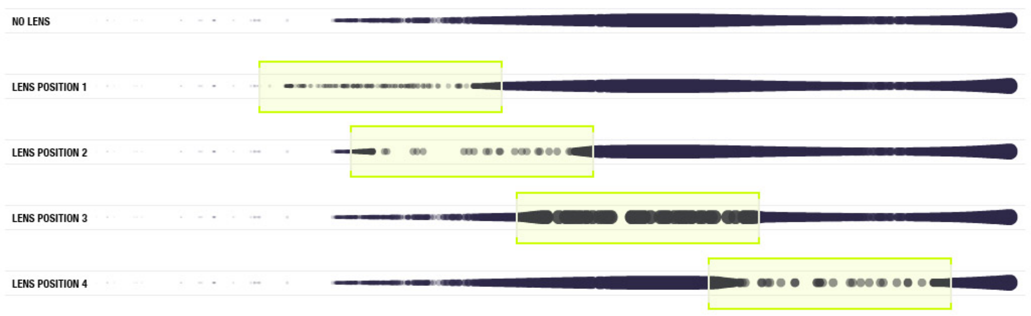


Figure 4.8: Magnifying the Barack Obama CloudLine. By positioning a lens tool on any part of the Barack Obama time interval, the analyst can see daily patterns, such as the gap in news coverage in position 2 or the convergence of coverage in position 3, and access individual news items. These potentially interesting patterns are impossible to see without magnification, as the no-lens line (top) illustrates.

and context space.

Lenses have been extensively researched in the human computer interaction and information visualization communities. Only recently high magnification lenses have been more thoroughly researched as a suitable Focus+Context technique in long time series [101]. In our approach, we employ a similar idea with two important differences: 1) our time series are event-based, not aggregated, and 2) we propose an improvement of a classical lens design that provides undistorted display of the focus region in a distorted space.

Similarly to SingalLens design [101], our lens design is based on a model

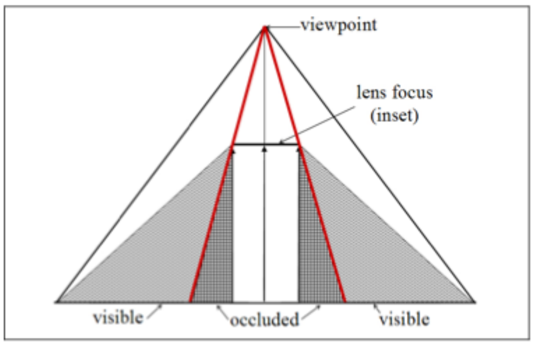


Figure 4.9: Lens schema from [33]

described by Carpendale and Montagnese in their Elastic Presentation Framework [32]. Each lens is described by a focus region of a width f, magnification factor m and drop-off areas d_f in which different functions can be used to give smooth transition between focus and context regions. In [33], Carpendale et al. have described different distortion functions used to achieve high magnification in Elastic Presentation Framework.

In CloudLines, we employ two different types of magnification lenses: 1) lens with linear magnification, and 2) lens with exponential magnification of logarithmic content.

4.4.2 Interaction Mechanics.

The size of the lens, as well as magnification level and the width of the focus region can be set by the user. When the lens is turned on, the user can inspect the time series by click-and-drag, and when the mouse button is released, the lens is left in its last position. When an object inside the focus region is selected, the respective event details are given on demand. If the user clicks outside of the lens area, the lens is repositioned and can be moved around the screen until this feature is turned off.

4.4.3 Details on Demand

Working with time series on item level is meaningful when each item carries information and this information has to be available for direct access for the user. For example, when working with news event data, where an event represents a time-stamped news article, each item can be enriched with different metadata information, such as the url of the article, title, image, etc. This requirement is reflected in our first decision to show a single event as an object that has a minimum size that is appropriate for interaction. Sometimes, more items can arrive at exactly the same time, and in these cases, the user should be provided with details on demand for all underlying data.

4.5 Case Study

To demonstrate our approach, we applied it to the news entities time series data collected by EMM news service [13] [107]. We present two examples that differ in volume and time range and show how CloudLines can be used in two different scenarios: 1) exploration of multiple long time series from one month of news event data; and 2) monitoring of 24 hours of news event data, and discuss the benefits and drawbacks of our solution.

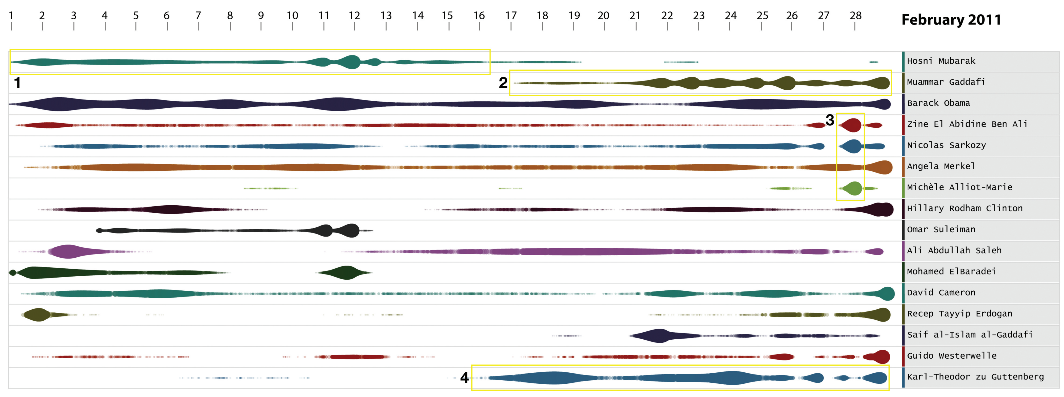


Figure 4.10: CloudLines applied on the news entities data collected during February 2011. Dataset consisting of around 117,000 events for 16 politicians with the highest volume during the whole month are shown (names are displayed on the right side). Compact visualization immediately reveals important event episodes. The fine structure of the data, such as the "pulse" of online publishing is also visible, showing higher activity in the afternoons. Detailed analysis of the visualization is given in subsection 4.5.1.

4.5.1 Exploration of Long News Sequences With CloudLines

The entity event data represent the most mentioned politicians during February 2011 and each event contains information about a news article that is mentioning a specific person (shown in Figure 4.10). When working with multiple time-series, finding an optimal way to sort the time-series is a difficult problem that goes beyond the scope of this work. The problem is addressed in detail in [193] and relates to the question of finding an optimal time series similarity measure. The volume of distinct time series can vary greatly and can be difficult to compare. When working with incremental data, this problem has to take into account re-sorting and has to allow for efficient computation. In our case, we have chosen 16 time series with highest overall popularity during the whole month and sorted them in a descending order.

Normalization. Time series can be normalized according to the different criteria, such as normalization to the overall volume of each time series, or to the peak value, etc. In Figure 4.10, the time series are normalized to the maximum peak value for each distinct time-series, since we would like to be able to see similar event episodes when they occur across different time series. Alternatively, the user can also choose to normalize the data to the maximum overall peak in our tool.

Looking at the visualization, several event episodes stand out and details can be revealed through interaction, which is described in the Section 4.4. After analyzing these interesting visual clusters, the highlighted and numbered regions in Figure 4.10 relate to the following events:

1. The crisis in Egypt during the first half of February 2011.

2. The crisis in Libya in the second half of February 2011.

3. A short event episode related to Michele Alliot-Marie, France's foreign minister and her ties to Tunisia's ousted president Ben Ali, after which she resigned from her position.

4. The (German defense minister) Guttenberg plagiarism scandal.

Additionally, the visualization reveals the fine structure of the data, such as a higher volume of published articles in the afternoon, describe the "pulse" of online news sources. The goal of the examples in our work is to demonstrate the validity of our technique for quick detection of important temporal event sequences in multiple long time series, although deeper analysis of each of these events is also possible.

The example shows one important feature of this type of event data: *the data varies greatly in shape and it can't be easily modeled as bursts* [102].

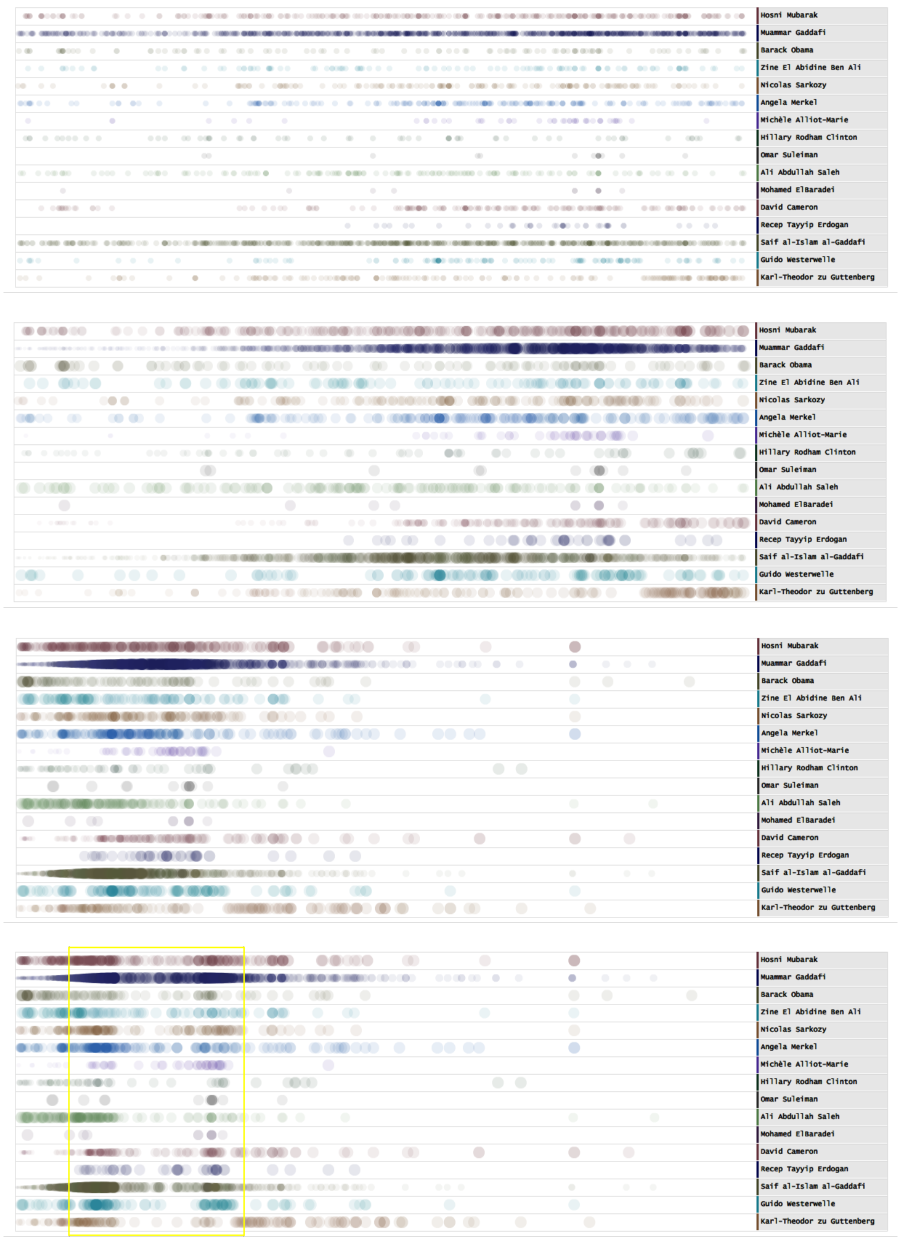


Figure 4.11: News monitoring scenario: Transformation of news entity event data in a 24 hour time window, taken on February 21, 2011. The data is appearing on the x-axis from left to right (12:00 AM is the boundary on the left, 11:59 PM is on the right). 6,756 events are shown. From top to bottom: 1) Raw data with high overlapping on a linear timeline; 2) Density based transformation; 3) Logarithmic distortion of the timeline gives more space for the latest events and compresses the past data into visual clusters; 4) Magnification lens applied on the past - focus region with lower density is now visible, and drop-off functions create more compressed areas that connect focus and context. Interaction with individual events is now possible.

4.5.2 News Monitoring with CloudLines

In our second example (shown in Figure 4.11), we are presenting the results of our approach on 24 hours of news event data in a monitoring scenario with the same choice of named entities as in the previous example. This time window is chosen for a scenario in which the user is aware of the current events, but needs to be able to follow any new incoming data. The visual density is much lower and allows the analyst to inspect individual events more easily.

The Figure 4.11 shows transformation of the visualized data with our method applied in consecutive steps, giving the final result in the last subfigure. Direct access to the new items is available immediately, and the user can interact with the newest events without magnification.

4.6 Summary

4.6.1 Discussion

The CloudLines technique could be applied to any type of irregular temporal data, where it is expected that the irregularities in the data distribution along the temporal axis could reveal interesting patterns. This section provides some additional remarks regarding the scalability and weaknesses of our approach. The techniques used in our approach can be adjusted to various application scenarios to satisfy different scalability and performance criteria. For the analysis of non-incremental data, offline density estimation has to be computed only once and the rendering could be optimized for highly overlapping events. In our first use case, around 117,000 events are processed and shown in Figure 4.10. To speed up the rendering, the overlapping events could be truncated and represented by a single object whose opacity would be the sum of underlying events, if their distance is less than a pre-defined ϵ, which depends on the loss of information that is acceptable for the user. The challenge, which is a part of our current research efforts, is seamless retrieval and re-rendering of truncated events in the focus region during user interaction. In our second case (Figure 4.11), which is focused on a shorter time window (24 hours), around 7,000 events are shown.

Kernel density estimators are known for having problems with tail regions of the distribution. This is especially problematic with the latest events, which will get undersmoothed due to the applied method. Second, density estimation is performed offline and re-run at user-specified time intervals. This fact currently prevents our design from being fully incremental for (near) real-time data stream monitoring. However, this problem can be overcome by applying an appropriate online density estimation algorithm and will be part of our future work.

The basic idea of our approach is to display multiple event sequences on atomic (event) level, i.e. without aggregation, and to allow *in situ* analysis of single events. The size of the event objects can introduce overplotting, especially in high density areas. The objects would have to be 1 pixel wide to minimize this problem, however the interaction would be practically impossible with such a design. There is no easy solution to overcome this problem for each and every dataset and limited screen space. Therefore, when high density areas (event episodes) are identified, the magnifying lens is used for direct inspection, thus removing overplotting in the focus region. In Figure 4.11, it can be seen that circles of the same size have different transparency. This is due to the fact that the events occur at the exact same points in time. The difference in size between circles strongly depends on the chosen kernel bandwidth and local density estimate. The effect of alpha-blending overplotting increases with circle size significantly and this perceptual issue is a limitation that we would like to address in our future work.

4.6.2 Conclusion

In this chapter, we have presented a novel interactive visualization approach for analysis of multiple event sequences in limited space. The idea behind CloudLines design is to explore the limits of large scale event data analysis on *atomic* level in limited space. Our first results show that our method with proposed Focus+Context techniques is worth further research. In our future work, we would like to evaluate our method with a formal user study. While we haven't conducted such a user study so far, the examples show the promising direction of our approach.

To demonstrate the usefulness and effectiveness of our approach, we have applied the techniques on a high-volume real world dataset in two extreme case examples. In our first example, we worked on one month of news entities event data and showed how the proposed technique can be used to:

- detect important event episodes in event sequences
- show the fine structure of the event episode shapes
- interact with event sequences on *atomic* level

In our second example, we used 24 hour of news entities event data and demonstrated how the technique could be used for real-time monitoring in short time frames. Additionally, our experiments showed that the proposed visualization method could also be suitable for a single event sequence. We believe that our approach can scale well with different data volumes, display resolutions and time ranges.

As part of our future work, we would like to improve the current lens magnification technique. Currently, we're using a cut-off function which truncates the data in order to remove the noise in the overview and increase performance. We allow the interaction in the regions of interest by using lens magnification as a Focus+Context technique. This technique could be improved by integrating a variation of semantic zoom, which would allow the retrieval of removed items at different zoom levels, based on their importance values.

5 Streaming Visual Analytics of Complex Story Development

Contents

Previous chapter presented a visual analytics approach for detection, analysis and exploration of temporal events in news streams. The fundamental assumption of the approach is that the event types are known in advance. In the examples, the event types were international politicians and the events were the news articles in which the politicians appeared in. In this chapter, I introduce a visual analytics approach designed for events whose type is not known in advance. The approach is realized as a system for exploration of *dynamic* events in news streams, *Story Tracker*, and it is based on the research framework presented in Chapter 2. Story Tracker combines interactive visualization and text mining techniques to allow exploration of complex stories, i.e. stories that split and merge over time.

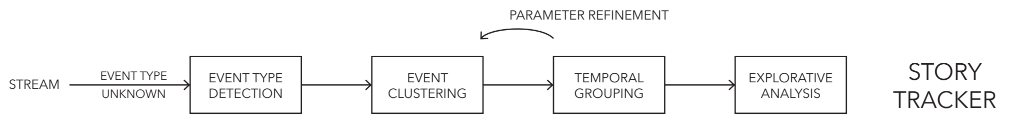


Figure 5.1: Story Tracker pipeline: The approach is applied on events whose type is not known in advance.

This chapter is based around the *story* model. Formally, each story is a group of similar text documents that are found in a document collection. Based on our definition of events, stories are event episodes that are dynamically created, i.e. they appear unexpectedly in the stream and disappear after some time. The basic flowchart of Story Tracker is shown in Figure 5.1.

In this chapter, the following contributions in particular are presented:

- An incremental visualization that shows evolution of stories that split and merge
- A sorting algorithm for multiple connected visual components that minimizes clutter and overlap from edge crossings
- An interaction technique for filtering the stories based on graph property *connectedness* and *duration*

- An analytical framework for text corpora that allows incremental updates of both topics and visualization, the refinement of clustering parameters and is based on the streaming visual analytics approach presented in Chapter 2.

The chapter is based on the following publication:

> *Milos Krstajic, Mohammad Najm-Araghi, Florian Mansmann and Daniel A. Keim. Story Tracker: Incremental Visual Text Analytics of News Story Development. Information Visualization, SAGE, 12(3-4):308-323, 2013. [109]*[1]

which is the extension of its predecessor:

> *Milos Krstajic, Mohammad Najm-Araghi, Florian Mansmann and Daniel A. Keim. Incremental Visual Text Analytics of News Story Development (Best Paper Award). SPIE 2012 Conference on Visualization and Data Analysis, 2012. [108]*[2]

5.1 How Do News Stories Evolve?

Understanding temporal development of unstructured and semi-structured text data streams is becoming increasingly important in many application areas, such as journalism, politics or business intelligence. At the same time, sources of textual data, such as online news providers are creating content in constantly growing amounts. While many automated and visual solutions that deal with the snapshots of information space already exist, few interactive systems can provide a user-friendly environment in which temporal context of these growing data collections can be successfully analyzed. The latest information coming from the news data providers prevails in the

[1] The paper is the result of the master thesis of Mohammad Najm-Araghi, who I have supervised at University of Konstanz. I have written the most of the publication, Mohammad Najm-Araghi implemented the system and wrote the case study, and Florian Mansmann gave advice and did the proofreading. Daniel Keim gave research advice.

[2] Ibid.

news landscape very quickly, putting the prior information out of the picture, even when it is necessary to keep the longer temporal context in mind to understand the current events. News stories that report on real-life events have characteristics that make analysis of this type of data challenging, from both the text mining and visualization perspective. Analyzing the temporal dynamics of the news content has been in focus of computer science researchers for some time. In the area of text data mining, a lot of effort has been put to model topics in evolving document collections. However, little work has been devoted to analyze complex relationships between news stories. In the visualization field, topic evolution is receiving more attention recently, although most of the proposed methods do not scale well when working with data streams. We believe that evolutive aspects of news stories cannot be separated from incremental visualization methods and that these methods must be coupled with interaction techniques that will allow the user to explore the rich content of text data streams in full detail.

We propose a novel news stream visual analytics system that integrates topic evolution algorithms with interactive visualization methods at three temporal zoom levels. To support effective analysis of the growing news corpora, we combine interactive methods with incremental visualization of story development to allow exploration of news data streams from a broader temporal overview to fine-detailed level of a single article that belongs to a particular story.

5.2 Incremental Detection of New Stories

The goal of my approach is to have a visual analytics system that can work with news streaming data, i.e. a sequence of time-stamped documents that arrive continuously over time. Theoretically, the volume and the speed at which the documents arrive are unbounded, and, therefore, it is desirable that the algorithms process the document corpora incrementally. In practice, news documents that are similar, i.e. report on a particular real-life event are usually temporally close. Therefore, it make sense to process the

WikiLeaks' Assange (28 documents)

(20) Assange: WikiLeaks to speed release of leaked docs
http://www.ynet.co.il/
(32) WikiLeaks' Assange faces new court hearing
http://www.euronews.net/

Hosni Mubarak (197 documents)

(46) Reports: Mubarak could be on his way out
http://www.upi.com/
(9) Egypt army steps in, sign Mubarak has lost power
http://hosted.ap.org/dynamic/fronts/HOME

World Cup (21 documents)

(229) IPL bags more FMCG, telco ads than World Cup
http://www.financialexpress.com/
(719) India has plenty of match-winners: Harbhajan Singh
http://www.rediff.com/

Table 5.1: Example of the output from the text clustering algorithm. The algorithm produces story titles, followed by the number of documents belonging to each story and a list of document IDs, titles and URLs. The articles are clustered in daily batches.

growing document collection at predefined time intervals in batches to create clusters of similar documents (as shown in Table 5.1), and then sequentially compare the clusters from neighboring intervals to find similar stories that span across a wider time frame. A set of documents in each time interval can be then treated as a static collection, which can be efficiently processed offline in order to find clusters of similar documents.

5.2.1 Detection of Daily Clusters

A conceptual diagram of the story cluster creation, detection of similar clusters and sorting mechanisms based on cluster size and strength is shown in Figure 5.2. The Story Creator module clusters the news articles in 24-hour time intervals. Since our news data stream source provides data from a large number of news sources, it is expected to have a lot of similar documents,

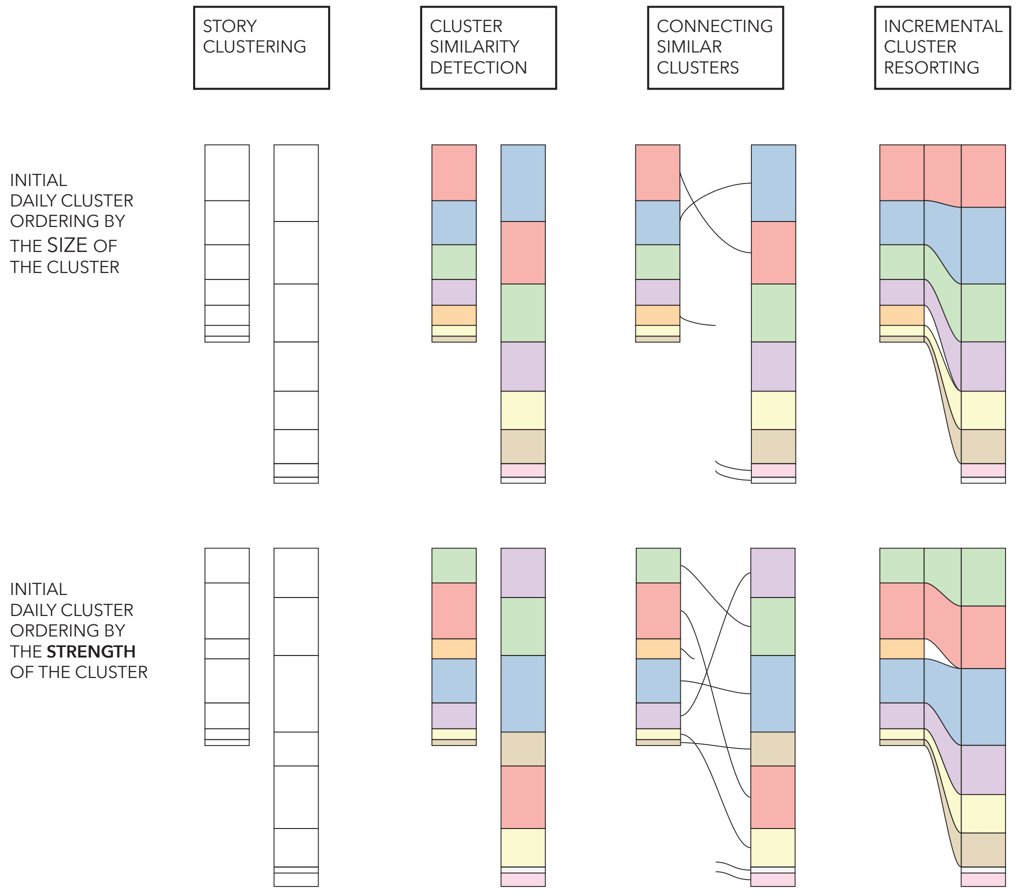

STORY CLUSTERING
CLUSTER SIMILARITY DETECTION
CONNECTING SIMILAR CLUSTERS
INCREMENTAL CLUSTER RESORTING
INITIAL DAILY CLUSTER ORDERING BY THE SIZE OF THE CLUSTER
INITIAL DAILY CLUSTER ORDERING BY THE STRENGTH OF THE CLUSTER

Figure 5.2: Story Tracker cluster detection, similarity identification, connection and sorting

which report on the same real-world event. Document clustering has been exhaustively researched in the field of text data mining, with several main directions, such as dynamic topic modeling based on LDA [26] and topic detection and tracking [8]. The evaluation of document clustering output can be quite exhausting and, although benchmark sets do exist, the results with our real news data were varying, depending on the selected time intervals and the amount of documents in the corpus. However, our visual analytics framework allows easy replacement of the underlying clustering technique. Furthermore, news topics are very often characterized by their braided nature [103], where topics overlap, split and merge. These characteristics can not be easily captured by existing topic modeling methods. In this work, we address this challenge by comparing similar clusters from adjunct time intervals to detect overlapping news stories.

The clustering module in our framework is based on Carrot^2 [130], an open source framework for clustering search results. This framework provides two clustering algorithms whose important advantage is twofold: first, they do not require a pre-defined number of clusters, and, second, they are very efficient in terms of processing time and computing power. The first algorithm is Lingo [129], which extracts frequent phrases from documents to produce high quality cluster descriptions (labels). Lingo is based on SVD (singular value decomposition) and uses VSM [151] to create term-document matrix, which is decomposed to create candidate labels and associate documents with the most similar labels. The second clustering algorithm is the Suffix Tree Clustering algorithm (STC) [195], which assumes that similar documents share identical phrases. In the world of news publishing, this is very often the case, since many online news portals publish modified versions of the original article, which was created by a global news agency, such as Reuters. Phrase-based approaches, such as STC, have the advantage over term-based clustering techniques, because the phrases are more informative than the representative (usually, most frequent) set of keywords and, additionally, they can be used to label clusters, which can be a problem with the other clustering techniques. STC has two phases: first, it creates *base clusters*, containing the sets of documents with an identical phrase, and, second, combines the base clusters to form *final clusters*. The assessment of the quality of clustering results performed by the STC algorithm can be found in the work by Stefanowski and Weiss [158].

For each news article, we use its title, summary, entities and tags (categories) as input for the clustering algorithm. The user can refine the input by including or excluding entities or tags in the interaction phase. A short example of a daily cluster output is shown in Table 5.1, with three identified stories: *Sidi Bouzid*, *Wikileaks' Assange* and *Japan*. The number of documents assigned to each story is given in the brackets, followed by the list of document IDs, titles and URLs.

5.2.2 Connecting the Dots - Comparing the Stories

Jan 1, 2011	**Jan 2, 2011**	**Jan 3, 2011**
Hosni Mubarak (371 documents) (0)US officials ask Egypt to hurry changes http://www.irishsun.com/ (3) Live blog: Essentially a military coup? http://www.msnbc.msn.com/ ...	**WikiLeaks' Assange** (28 documents) (20) Assange: WikiLeaks to speed release of leaked docs http://www.ynet.co.il/ (32) WikiLeaks' Assange faces new court hearing http://www.euronews.net/ ...	**Sidi Bouzid** (118 documents) (19) Tunisian Government: 14 Killed as Rioting Continues http://www1.voanews.com/english/news/ (26) Tunisia 'to respond' to protests http://english.aljazeera.net/English …
Tahrir Square (271 documents) (4) Mubarak meets with VP, protesters flood square http://www.ynetnews.com/home/ (6) Will Mubarak step down today? Protesters told demans http://www.thestar.com/ ...	**Hosni Mubarak** (197 documents) (46) Reports: Mubarak could be on his way out http://www.upi.com/ (9) Egypt army steps in, sign Mubarak has lost power http://hosted.ap.org/dynamic/fronts/HOME ...	**Cricket World Cup** (121 documents) (403) Irish skipper Porterfield confident http://www.antiguanews.com/ (416) India will feel pinch from last loss, says Bangladesh opener http://www.irishsun.com/ …
Wall Street (134 documents) (114) Facebook, Google eye Twitter takeover http://www.expressindia.com/ (169) Facebook, Google in Twitter takeover talks: WSJ http://timesofindia.indiatimes.com/ ...	**World Cup** (21 documents) (229) IPL bags more FMCG, telco ads than World Cup http://www.financialexpress.com/ (719) India has plenty of match-winners: Harbhajan Singh http://www.rediff.com/ ...	**Japan** (114 documents) (5) Japan, American Airlines alliance being boosted http://thestar.com.my/ (136) Japan, American Airlines Alliance Being Boosted http://www.irishsun.com/ …

Table 5.2: Comparison of daily clusters. For each day, a list of output stories and documents that are assigned to them is produced. The title words, descriptions and cluster labels are compared to detect stories that span over more than one day.

Major stories can span over a longer period of time and understanding the evolution of these stories is one of the challenges that we are dealing with in our work. Since we want to be able to process data incrementally, the stories discovered in each time interval (24 hours), can be compared to the stories from the previous n intervals. Rose et al. [145] use $n = 7$ in their evaluations. In our case, we compare the stories from the current and the previous day, which helps in dealing with visual clutter that arises when $n > 1$. Besides, our experiments showed that, in most cases, stories appear on consecutive days without long breaks between them.

Essential content of each story consists of a set of keywords coming from the story title, description and document title words. We use Jaccard distance [85] to calculate the similarity between stories belonging to neighboring time intervals, which is computed on the extracted set of keywords. An example of the clustering output from three consecutive days is shown in Table 5.2. The highlighted and colored keywords in the titles show the high similarity between articles and stories.

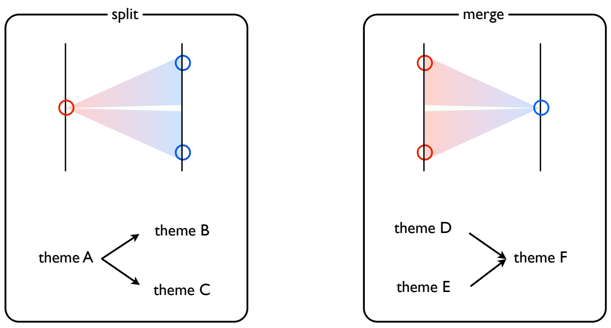


Figure 5.3: Splitting and merging of news stories. Story A can split into stories B and C (left)*. Similarly, two originally disjoint stories D and E can merge into one main story F* (right)*.*

5.2.3 Merging and Splitting

Evaluation of document clustering techniques is a daunting task. Very often, there is no clear cut between stories, and documents that are split into different clusters can be still very related by their content. Furthermore, news stories may disintegrate into two different topics during their evolution over time, or they can dissolve into one single topic (as shown in Figure 5.3). To address this issue, we have empirically set two thresholds: one for the splitting and one for the merging of news stories. The calculated cluster similarity values are then used for connecting both the most similar stories between consecutive days and also the stories whose similarities are higher than the given threshold. Therefore, when a story splits, each "child" story cluster that evolves from the "parent" story cluster will have the same visual encoding as the "parent". In case of merging of two stories into one, the new story will inherit visual features (namely, color) from the most similar story from the previous day. The threshold value can be later adjusted by the user to create more tightly or more loosely connected stories. The details about visual encoding of stories and their evolution are given in the section Visualization of Detected Stories Using the Streaming Visual Analytics Approach.

5.3 Visualization of Detected Stories Using the Streaming Visual Analytics Approach

Following the design considerations and streaming visualization properties presented in chapter 2, three basic requirements that our design needs to fulfil in order to support explorative analysis of story development are:

1. The user needs to understand the passage of time *(providing temporal context)*.

2. The user has to be able to differentiate the objects that evolve over time by their importance (volume, rate of change, temporal patterns etc) *(updating strategy)*.

3. The visualization should be able to accommodate new data, without disrupting the user's mental map of the past *(layout stability, incrementality)*.

Well-known techniques that have been used in the past to visualize multiple numerical time series, such as horizon graphs [78], simple line charts or temporal heatmaps have a major disadvantage that they require a fixed vertical size for each (pre-defined) visual object (in our case, a story). However, the stories appear and disappear over time and we do not know in advance how long a certain story will last. Therefore, these techniques are expensive in terms of real estate space and inefficient. Another popular technique, ThemeRiver [77], allows a more efficient screen-space usage for new objects in the visualization. However, the technique is not incremental, i.e. the positioning of the objects (layers) has to be recalculated every time new data is added. We have developed a hybrid technique, which combines the advantages of connected ordered lists and themeriver-like visualizations. This allows the user to balance-out disadvantages using interaction.

For designing a visualization of news story evolution over time, we have to take into account the following analytical and visualization issues: (a) the user needs to be able to identify, track, and analyze stories and understand

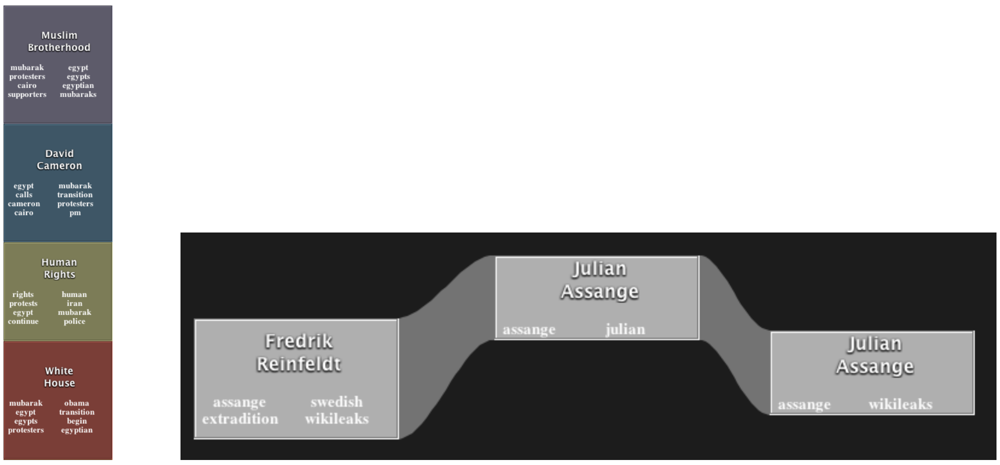

Muslim Brotherhood
mubarak protesters cairo supporters
egypt egypts egyptian mubaraks
David Cameron
egypt calls cameron cairo
mubarak transition protesters pm
Human Rights
rights protests egypt continue
human iran mubarak police
White House
mubarak egypt egypts protesters
obama transition begin egyptian
Fredrik Reinfeldt
assange extradition
swedish wikileaks
Julian Assange
assange
julian
Julian Assange
assange
wikileaks

*Figure 5.4: Basic visual elements in the visualization: a daily cluster representation (*left*) and a story evolution flow over three consecutive days (*right*).*

their relative importance without reading them; (b) the visual stability of the layout has to be maintained, with minimal clutter. These two basic criteria are conflicting in certain aspects. Therefore, the user needs to be able to control the visualization output and understand the consequences of his interaction steps.

The text processing module of our system analyzes the documents in batches of pre-defined intervals and produces clusters of documents that represent news stories report ing on a particular real-life event. Each news story is described by a story title (label), most important keywords, people and organizations mentioned in the news, but also by the *strength* of the cluster and the number of documents that belong to the story. Figure 5.4 shows the basic visual elements of our system consisting of cluster representations and the story evolution flow to connect semantically related clusters of subsequent days.

5.3.1 Visualizing a Month of Stories

In order to provide a broader temporal context, we designed a *Monthly View*, which depicts less detail about particular stories, but gives an overview in which the analyst can get the first idea about the size of data, the evolution of the news stories and their relationships. This view, shown in Figure 5.5, is enhanced by interactive filtering, which is described in more detail in section 5.4.

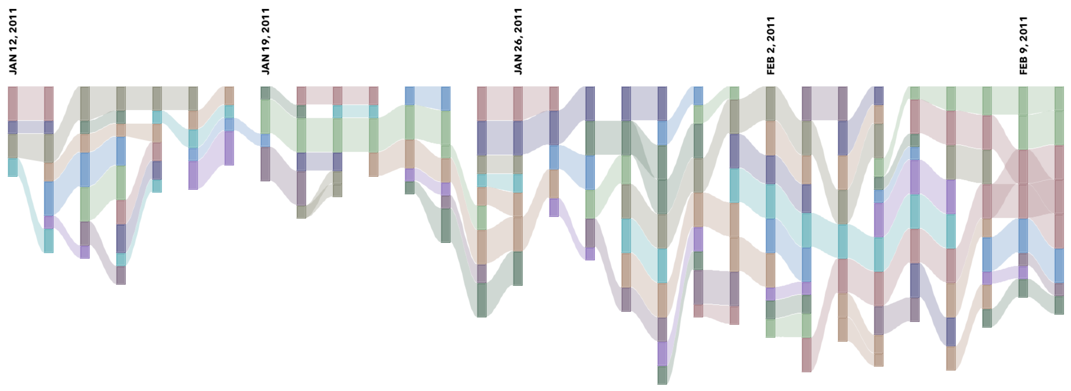


Figure 5.5: Overview of persistent news themes from January 12 to February 10, 2011. The themes can be filtered by duration and theme connectivity using filtering capabilities of the system. Additionally, the user can reduce clutter in the news landscape by minimizing edge crossings.

5.3.2 Main View

The fundamental component of our visual design is a representation of the output for a specific time interval, as shown in the *Main View* in Figure 5.6. The visualization places days along the horizontal axis and daily stories are stacked in the single column list and ordered within the list by the strength of the story cluster. Each story is represented by a rectangle in the list, whose height is mapped to the number of documents assigned to the story. This allows the story title and the most important keywords to be displayed inside the rectangle. To provide an overview of the temporal evolution of the stories, we are using two visual clues: a story that evolves during several days is connected with shapes based on Bezier curves and it is colored, while the story that appears for only one day remains grey, which serves a dual function: first, the user can easily distinguish between longer and one-day stories, and, second, the flow of the story can be followed more easily when the interpolating shapes of several different stories overlap. The colors used

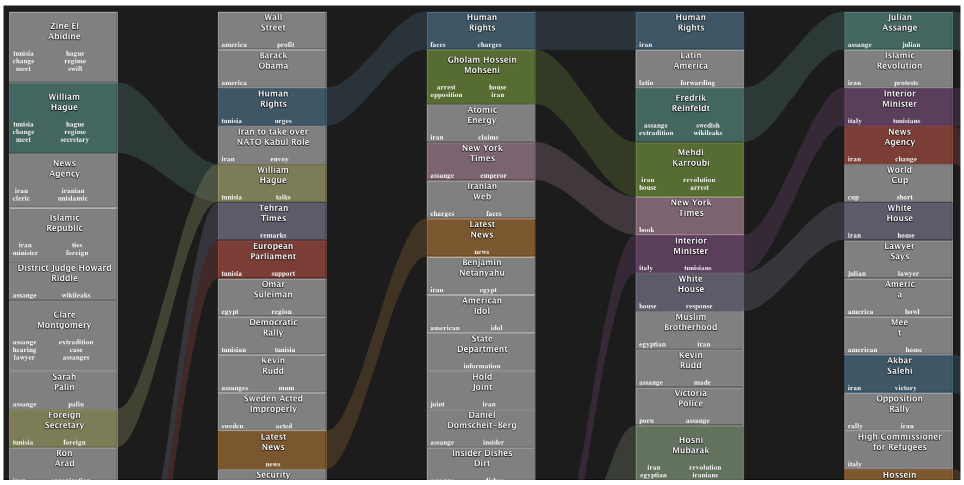


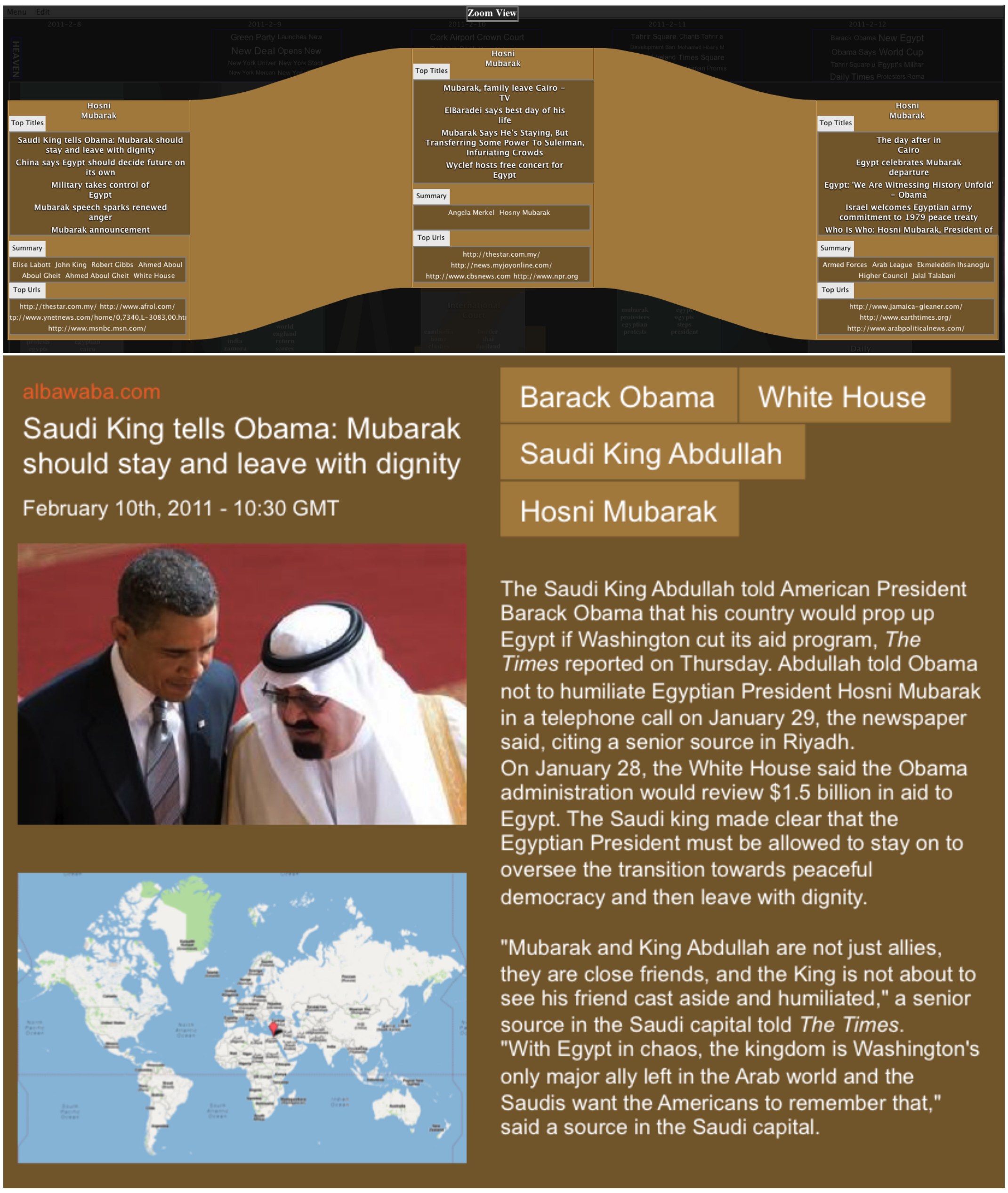
Figure 5.6: Main View: Visualization of news stories with default parameter settings. The stacks of rectangles show daily news clusters, where stories that span across several days are connected and colored, while the stories that appear only on a single day are grey. The stories are, by default, sorted by cluster strength, and the size of the rectangle corresponds to the number of article in a story. The rectangles are labeled with the story title and the most important keywords.

in our system are selected using ColorBrewer [76].

This representation gives a good balance between mid-range temporal evolution, level of detail and importance of news stories. Navigation in this space is possible both in horizontal (time) and vertical (story importance) direction.

5.3.3 Zoomed View and Article View

To explore the contents of a particular story in detail, we developed the *Zoomed View*, which gives detailed information about the documents assigned to the story as details on demand. Figure 5.7 shows Zoomed View for 3 days of a story titled Hosni Mubarak, containing titles of articles that belong to the story, summary and top URLs for each day. The *Article View* provides detailed information about each article, including full text, related entities and images. If geographic location is detected in the text, the map is also displayed.



Hosni Mubarak

Saudi King tells Obama: Mubarak should stay and leave with dignity
China says Egypt should decide future on its own
Military takes control of Egypt
Mubarak speech sparks renewed anger
Mubarak announcement

Elise Labott John King Robert Gibbs Ahmed Aboul Aboul Gheit Ahmed Aboul Gheit White House

http://thestar.com.my/ http://www.afrol.com/
tp://www.ynetnews.com/home/0,7340,L-3083,00.ht
http://www.msnbc.msn.com/

Hosni Mubarak

Mubarak, family leave Cairo – TV
ElBaradei says best day of his life
Mubarak Says He's Staying, But Transferring Some Power To Suleiman, Infuriating Crowds
Wyclef hosts free concert for Egypt

Angela Merkel Hosny Mubarak

http://thestar.com.my/
http://news.myjoyonline.com/
http://www.cbsnews.com http://www.npr.org

Hosni Mubarak

The day after in Cairo
Egypt celebrates Mubarak departure
Egypt: 'We Are Witnessing History Unfold' – Obama
Israel welcomes Egyptian army commitment to 1979 peace treaty
Who Is Who: Hosni Mubarak, President of

Armed Forces Arab League Ekmeleddin Ihsanoglu Higher Council Jalal Talabani

http://www.jamaica-gleaner.com/
http://www.earthtimes.org/
http://www.arabpoliticalnews.com/

albawaba.com

Saudi King tells Obama: Mubarak should stay and leave with dignity

February 10th, 2011 - 10:30 GMT

Barack Obama White House
Saudi King Abdullah
Hosni Mubarak

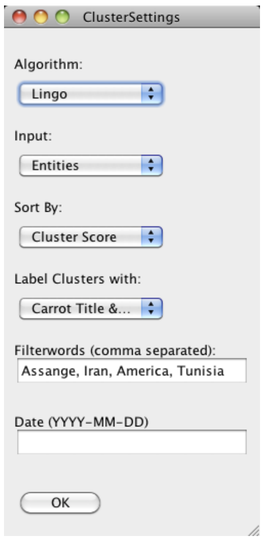
The Saudi King Abdullah told American President Barack Obama that his country would prop up Egypt if Washington cut its aid program, *The Times* reported on Thursday. Abdullah told Obama not to humiliate Egyptian President Hosni Mubarak in a telephone call on January 29, the newspaper said, citing a senior source in Riyadh.
On January 28, the White House said the Obama administration would review $1.5 billion in aid to Egypt. The Saudi king made clear that the Egyptian President must be allowed to stay on to oversee the transition towards peaceful democracy and then leave with dignity.

"Mubarak and King Abdullah are not just allies, they are close friends, and the King is not about to see his friend cast aside and humiliated," a senior source in the Saudi capital told *The Times*.
"With Egypt in chaos, the kingdom is Washington's only major ally left in the Arab world and the Saudis want the Americans to remember that," said a source in the Saudi capital.

Figure 5.7: The Zoomed View (top) provides fine-detailed analysis of each story. Each daily cluster provides the titles of top 5 documents, short summary of the cluster and the most important URLs. The Article View (bottom) shows the retrieved text of the article with additional metadata for the story titled "Saudi King tells Obama: Mubarak should stay and leave with dignity".

5.4 Explorative Analysis and Interaction

As a first step in the analysis, the user can change the input data parameters and the criterion for sorting of the daily clusters using the settings window shown in Figure 5.8. First, the algorithm, which is used for news clustering, can be changed between STC [195] and Lingo [129]. This feature can be easily extended to allow other clustering algorithms. Second, the user can select the type of metadata that is used as the additional input: *named entities* and/or *news tags*. The named entities are found in the news articles using named entity recognition algorithms, while the tags are specific categorical keywords that describe the article. Both metadata types are used as weighted keywords to improve the clustering results. Changing the parameters allows the analyst to understand the differences between different parameter settings and the influence of metadata on the output.



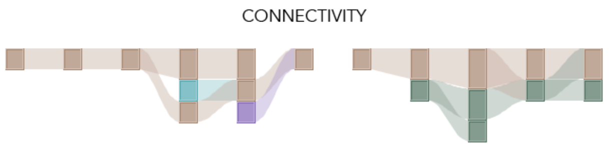
Algorithm: Lingo
Input: Entities
Sort By: Cluster Score
Label Clusters with: Carrot Title &...
Filterwords (comma separated): Assange, Iran, America, Tunisia
Date (YYYY-MM-DD)

Figure 5.8: Input parameters: clustering algorithm, metadata, sorting criteria, cluster labels, filtering keywords and time range.

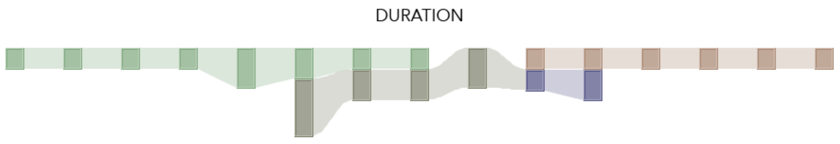
Furthermore, the user can select sorting criteria, either by the cluster strength or by the number of articles in the cluster. By default, the daily story clusters are ordered by cluster strength score, which determines how tightly the documents that belong to the cluster are connected. Alternatively, the user can choose to sort the clusters by the number of documents that belong to the cluster, which can be considered as another measure of cluster importance. The labels of daily clusters can be created either by the automated labeling algorithm or by using the most important article from the cluster.

Confirmatory analysis is supported by allowing the user to add filtering keywords and a specific time range to focus on a certain part of the dataset.

The explorative analysis of the news landscape starts with the monthly overview, followed by the main (weekly) view and the zoomed view (for details on demand). The visualization of the full month of data is designed to provide a broad overview of the dataset and give the user a basic insight into the major stories and the temporal patterns. Therefore, the textual labels are omitted in this view. Depending on the daily volume and the time range, the visualization can become easily cluttered with daily cluster connections and quickly changing order of the stories within the daily listings. A large number of short-lived stories, which might not be relevant to the analyst at this level, aggravate the problem. Since we are developing a system that supports incremental processing and visualization of a data stream, we need a solution that will maintain the layout of the past data regardless of the amount of new information. In order to address these challenges, we developed a filtering mechanism that allows the user to minimize the clutter and detect interesting patterns in this view.



(a) Connectivity filter removes stories with low similarity.



(b) Duration filter keeps long stories.

Figure 5.9: Two filters are used to clean up the clutter in the Monthly View: connectivity and duration

5.4.1 Reducing Overlap and Clutter: Filtering and Sorting Algorithm

The problem of inter-interval connections clutter can be regarded as a graph layout problem. The daily clusters are the graph nodes and the connections between the clusters are the edges directed from the past to the present.

The user can use the *connectivity* and *duration* filters to remove weakly connected and short stories. Using connectivity filter (Figure 5.9a), we can filter out the stories that do not split or merge and therefore keep only the stories with the most braided characteristics on the screen, while the duration filter (Figure 5.9b) allows us to keep the stories that span over multiple days.

To minimize the clutter which is due to the daily cluster connections, we have to find an optimal solution for minimization of edge crossings. Our goal is to optimize a directed graph, where the direction of the edges represents a hierarchy. The hierarchical layout should:

1. have as few edge crossings as possible
2. be presented vertically and in a straight-path
3. have uniformly distributed nodes and avoid long edges

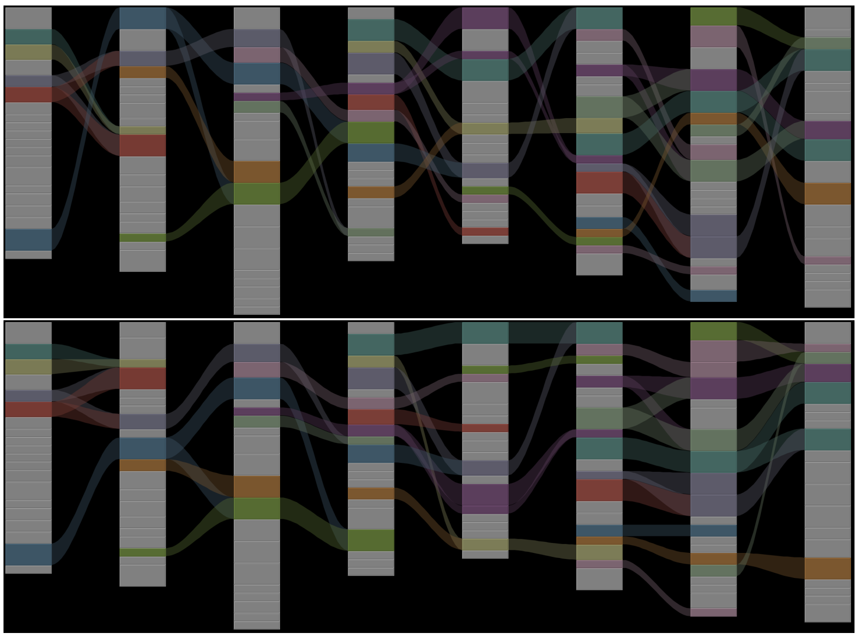


Figure 5.10: Default sorting (top) and minimized edge crossings (bottom)

As an additional remark, we have to consider that the first condition can only be achieved by acyclic graphs, which we have in our case. A possible method for such a layout is based on the Sugiyama algorithm [162]. This method consists of four steps, which are difficult optimization problems:

1. *Delete all cycles*: Find a minimal number of edges according to the distance to which the graph is acyclic and switch their direction.

2. *Layer positioning*: Calculate a good allocation of the edges in all layers, so that they are directed upwardly. Replace all edges that go beyond one layer. (This step can be skipped, because of our layout.)

3. *Minimize the edges*: Calculate for each layer a layout so that the number of crossings is minimal.

4. *Positioning*: Calculate the x-coordinates of the nodes in this way that we have no overlap. (Since our coordinates are fixed, this step can be skipped.)

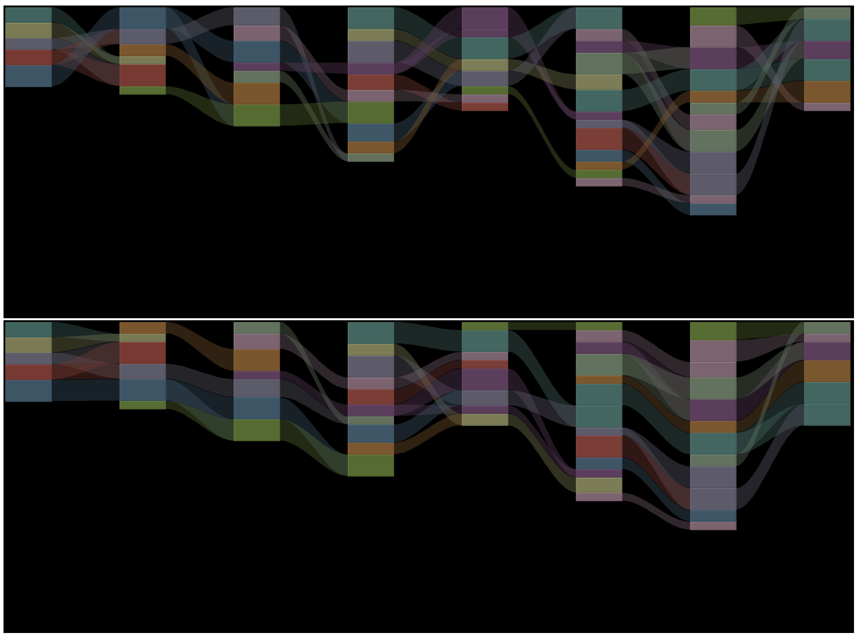


Figure 5.11: Connectivity filter with (bottom) and without (top) edge crossings minimization.

The third phase is NP hard, because changing one layer affects the next layer. To ensure that the story order for each day is unaffected by the future, we have reduced the problem to a two layer crossing minimization.

This preserves the organization of the topics in the story flow. Therefore, the heuristic starts with the second layer. After sorting the nodes in this layer, we proceed iteratively with the next layer. The results of applying edge crossing minimization are shown in Figure 5.10 (without filtering) and Figure 5.11 (with connectivity filter applied).

5.4.2 Additional Visualization and Interaction Features

Different clustering methods, data input selectors and sorting criteria allow the user to control the amount of data that will be visualized with our tool, where the data can be shown with three levels of detail (monthly view, main view and zoomed view). Still, the complex properties of news articles and the conflicting criteria of importance-based sorting and visual stability can make it difficult to track a particular story. Therefore, we have also implemented highlighting of the stories on mouse hovering and tooltip information, which can be seen in Figure 5.12.

In the real-world analysis scenario, active exploration of the news landscape using presented interaction methods and filters can change the order and positioning of the stories. To help the user in understanding the changes that different settings cause to the visualization, the animation is used to smoothly transition between two states.

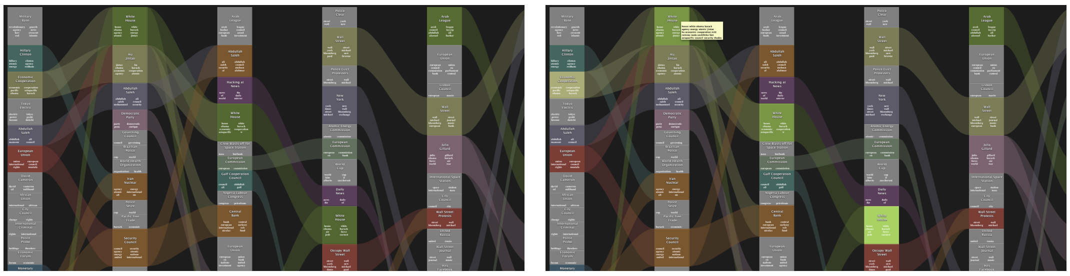
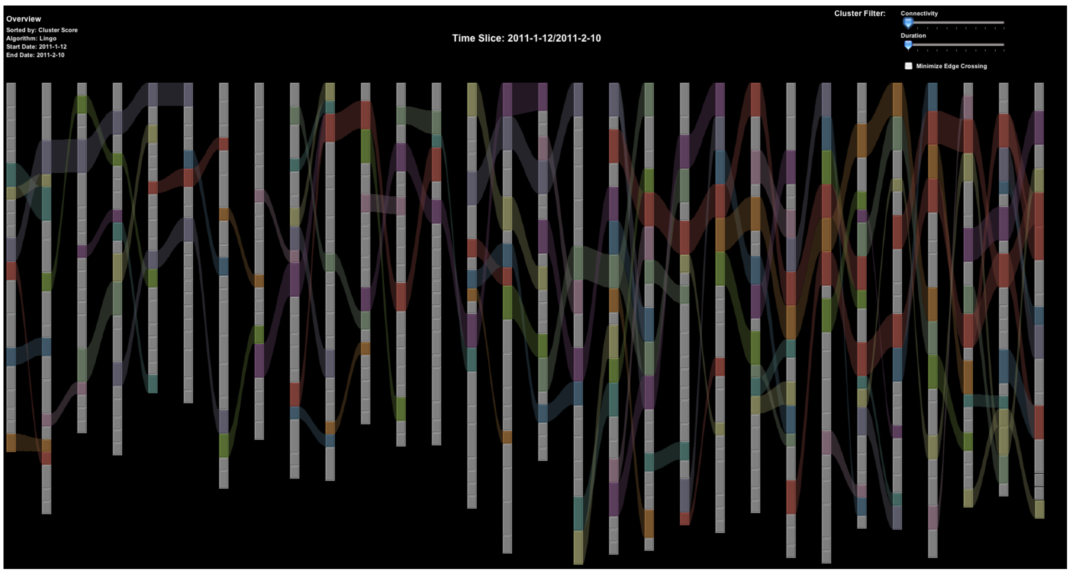


Figure 5.12: Highlighting with tooltip information.

5.5 Use Case: the Arabic Uprising 2011 and user study


Overview
Sorted by: Cluster Score
Algorithm: Lingo
Start Date: 2011-1-12
End Date: 2011-2-10

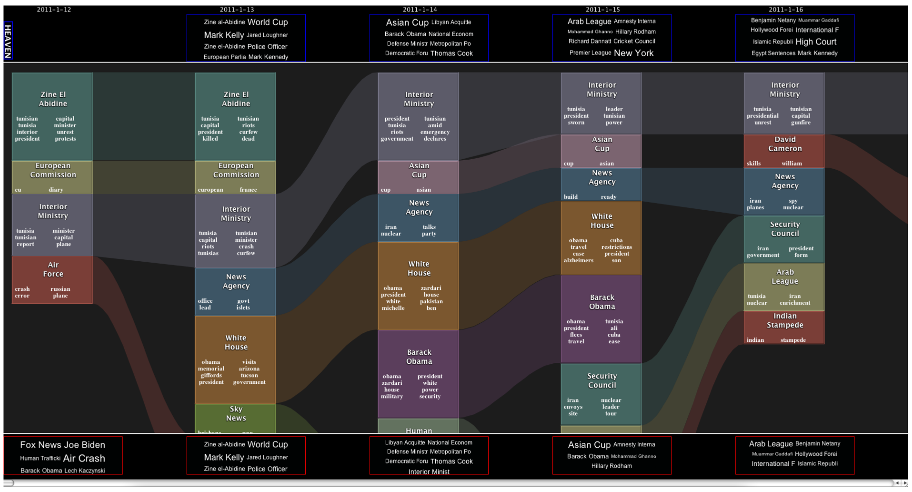
Time Slice: 2011-1-12/2011-2-10

Figure 5.13: Overview of the news themes from January 12 to February 10, 2011. The themes can be filtered by duration and theme connectivity using filtering capabilities of the system. Additionally, the user can reduce clutter in the news landscape by minimizing edge crossings.

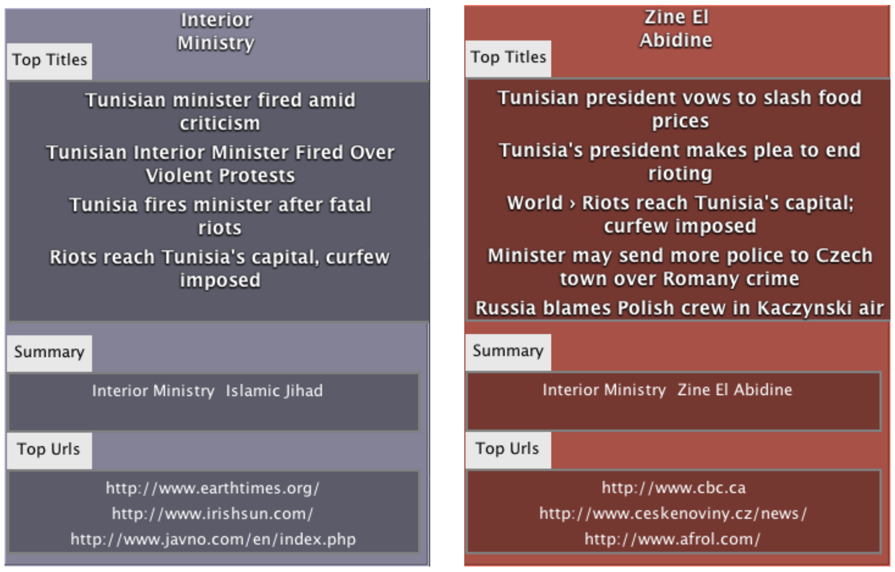
Our system allows interactive analysis of the large international news landscape with the focus on temporal development of news topics. To demonstrate the added value that our tool gives to a news analyst, we focused our use case on the Arabic uprising in 2010 and 2011. Since December 18, 2010, a revolutionary wave has spread over more than 15 Arabic speaking countries. The Tunisian Revolt was the starting point and devolved, step by step, across other Northern African countries. Even the Arabian Peninsula was not excluded from minor demonstrations up to governmental changes.

5.5.1 The Tunisian Riot

Figure 5.13 gives an overview of the stories detected in the news between January 12 and February 10, 2011. Through filtering and edge minimiza-



(a) Filtered Main View



Interior Ministry

Top Titles

Tunisian minister fired amid criticism

Tunisian Interior Minister Fired Over Violent Protests

Tunisia fires minister after fatal riots

Riots reach Tunisia's capital, curfew imposed

Summary

Interior Ministry Islamic Jihad

Top Urls

http://www.earthtimes.org/
http://www.irishsun.com/
http://www.javno.com/en/index.php

Zine El Abidine

Top Titles

Tunisian president vows to slash food prices

Tunisia's president makes plea to end rioting

World › Riots reach Tunisia's capital; curfew imposed

Minister may send more police to Czech town over Romany crime

Russia blames Polish crew in Kaczynski air

Summary

Interior Ministry Zine El Abidine

Top Urls

http://www.cbc.ca
http://www.ceskenoviny.cz/news/
http://www.afrol.com/

(b) Semantic Zoom on stories titled Interior Ministry and Zine El Abidine

Figure 5.14: Story Tracker case study: Tunisian Revolt – The filtered main view from January 12, 2011 until January 16, 2011 reveals stories related to the Tunisian Interior Ministry and the Tunisian president Zine El Abidine Ben Ali. Details on demand show top titles, brief summary and most important URLs for each daily story cluster

tion, the otherwise complex media landscape becomes easier to interpret due to the fact that many short unrelated themes are discarded and the strict theme ordering criteria according to popularity are relaxed. To avoid a loss of possible interesting stories, it is, for example, recommended to filter out all one-day stories by moving the slider at the top right. The next useful filtering step is to select the 'minimize edges' checkbox. From now on, the user can follow the flow of stories over 30 days, since positioning does not only depend on a theme's popularity for each day, but also on the theme's position on the previous day. The filtered view with persistent themes can be seen in Figure 5.5.

After interactive filtering and automatic layout optimization, the user can follow the ongoing stories in the main view. By looking at the top keywords and entering the zoomed view, the user can identify from the main view, shown in Figure 5.14a, that the green and purple-grey colored themes cover the riots in Tunisia. The growing importance of these events in the news is coherent to the reports in the Wikipedia's *Current Events* portal[3]. The portal lists all occurrences in one month and also provides a snippet with a short summary of each event. For the 12th of January 2011, the snippet about the protests states the following: "Tunisia's Interior Minister Rafik Belhaj Kacem is sacked by President Zine El Abidine Ben Ali, who also orders the release of most people detained during recent unrest". At this point in time, Zine El Abidine Ben Ali, the President of Tunisia, is in the focus of the event. From now on, the suspension of the Interior Minister becomes an important event and the main view absolutely conforms with the political events. To discriminate between the events that belong to this long story, keywords and phrases are displayed. In this case, 'minister' or 'president' are characteristic terms. Nevertheless, these topics are still connected, because the protests were against the entire government, including the ministers and Ben Ali.

For detailed analysis, the semantic zoom shown in Figure 5.14b reveals more information about both themes. This level of detail shows us clearly what we already briefly identified in the main view. The documents, which were assigned to the 'Interior Minister' topic, refer to the event which was mentioned before. The first three titles are about the sack of Rafik Belhaj Kacem in the context of violent protests. In contrast, Zine El Abidine con-

[3]http://en.wikipedia.org/wiki/January_2011

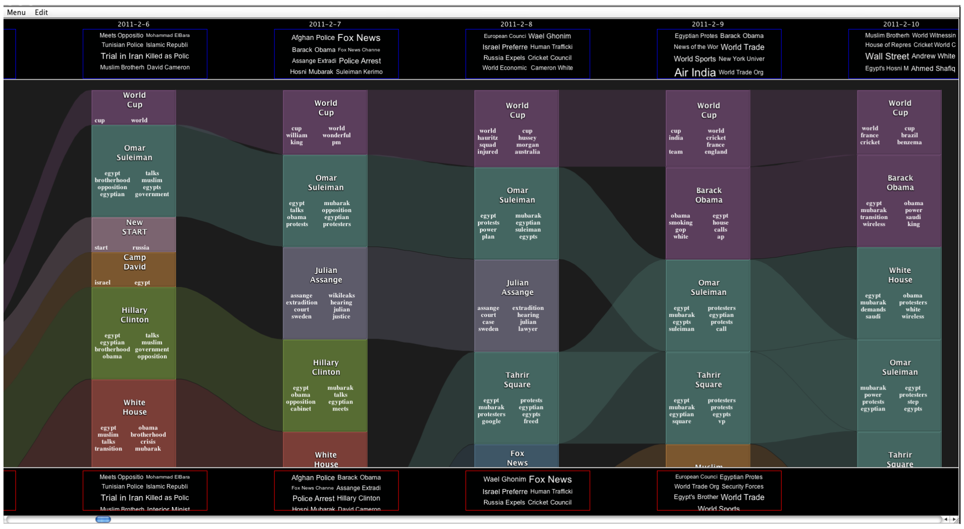
tains documents which refer to himself. As we can recognize in Figure 5.14b there is still some noise included - the fourth and fifth document do not belong to the topic but were highly ranked by the ″most important title″ algorithm.

In the second week of the given time window, Tunisia and Zine El Abidine are still a major topic in the news. The *Current Events* portal reports about Tunisia's army firing on the citizens and governmental changes and the dismissal of more ministers from the Constitutional Democratic Rally party that had governed the country. This textual information, represented in a list view where no context is visible, is mapped as an ongoing stream in the main view. Zine El Abidine remains a representative main phrase, and self-explanatory keywords like 'government', 'quit', 'minister', 'fallen', 'victims' and 'revolution' mirror the actual state.

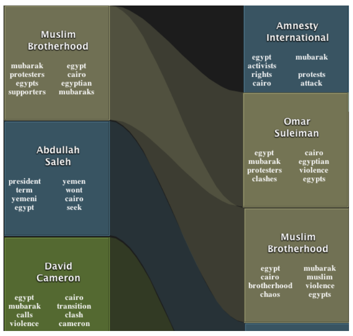
5.5.2 The Riots in Egypt

Figure 5.15a shows a filtered view from the 6th until the 10th of February 2011. We can easily identify that the news were dominated by the riots in Egypt. Several different stories reporting on the same topic appear in this view. Omar Suleiman, a former Egyptian army general, and Hosni Mubarak's Vice President, play an important role here. The reason can be identified by reading the top title words. The daily cluster shapes contain the terms about the 'Muslim Brotherhood' and a possible opposition. At this stage, Omar Suleiman and the Brotherhood were discussed as possible successors of Hosni Mubarak. Other protagonists, like Hillary Clinton, or Barack Obama, were also involved in the discussions. Besides, different topics like the cricket World Cup and the story about Julian Assange appear among the most important.

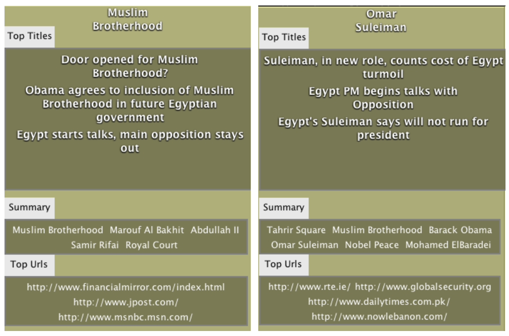
By scrolling the main window horizontally back into the past, the user can see that the Tunisia and Yemen protests dominated the streams and built up a large number of stories, similar to what we discovered during the riots in Egypt. We selected a split of one theme into several ones as one of many interesting patterns of the news flow to demonstrate the effectiveness of our



(a) Filtered main view from Feb 6, 2011 until Feb 10, 2011



(b) Splitting of a single story into two on the next day



Muslim Brotherhood

Top Titles

Door opened for Muslim Brotherhood?

Obama agrees to inclusion of Muslim Brotherhood in future Egyptian government

Egypt starts talks, main opposition stays out

Summary

Muslim Brotherhood Marouf Al Bakhit Abdullah II Samir Rifai Royal Court

Top Urls

http://www.financialmirror.com/index.html
http://www.jpost.com/
http://www.msnbc.msn.com/

Omar Suleiman

Top Titles

Suleiman, in new role, counts cost of Egypt turmoil

Egypt PM begins talks with Opposition

Egypt's Suleiman says will not run for president

Summary

Tahrir Square Muslim Brotherhood Barack Obama Omar Suleiman Nobel Peace Mohamed ElBaradei

Top Urls

http://www.rte.ie/ http://www.globalsecurity.org
http://www.dailytimes.com.pk/
http://www.nowlebanon.com/

(c) Zoomed Views of the stories on Muslim Brotherhood and Omar Suleiman

Figure 5.15: Story Tracker case study: The riots in Egypt

system in describing story splitting patterns. As Figure 5.15b reveals, the Muslim Brotherhood was displayed as a root theme, which leads to Omar Suleiman as a new ongoing theme.

At that point, severe discussions about the opposition and possible successors broke out. Both the Muslim Brotherhood and Omar Suleiman were part of these discussions. Figure 5.15c shows the result of applying the semantic zoom on the two themes. The documents assigned to each topic help to understand the causes for this split. The user can identify individual titles like *Door opened for Muslim Brotherhood* and *Suleiman, in new role, counts cost of Egypt turmoil*. These titles discriminate clearly between topics.

However, the discussion about the opposition and the possible successors are connecting points for these events. This mixture causes a split which is in this case justifiable. As described in the section Incremental Detection of New Stories, the splitting and merging of themes is dependent on previously defined similarity threshold. To be more concrete, a split will be defined if the calculated Jaccard similarity coefficient is less than 30%. However, depending on each concrete use case, this parameter needs to be fine-tuned since it is possible that a split or merge is identified, while it should not, or the other way around.

5.5.3 User Study

We recruited 8 users to test our prototype in order to get an initial feedback on our approach in an informal user study. The participants were experts from social sciences (five) and computer science (three). At the beginning, they were provided a short introductory session, in which the visualization was explained first and then the users were free to explore the interaction capabilities of the system. This phase took about 15 minutes, depending on the questions the users had about the data, visualization and interaction techniques. Our aim was to understand how the users perform the following tasks: (1) finding important stories (2) finding the most interrelated stories, and (3) ease of interaction with the system. We encouraged the participants to openly comment and express their opinion on the system and share their findings during the study. At the end, we conducted an interview to get their feedback about the overall experience with the tool.

The user study led to the following observations:

Overview first. The users spent considerable amount of time performing analysis in the Overview visualization. They used this view to filter out the short stories, reorder the long stories and, in general, to get the basic understanding of the whole dataset. They used this view to significantly reduce the dataset to only the most important stories and then switched to the Main View.

Cluster labels. All participants complained that the daily cluster labels were not informative enough when the Lingo algorithm was used. They also suggested better use of space within each daily cluster box in the Main View, by adding more detailed cluster descriptions.

Use of color. In the beginning, some participants felt misled by the use of the same color for different stories. After getting used to the color mapping, they did not complain about this feature.

Our limited user study showed us that our approach was well-received within the initial user group, regardless of their professional background. However, we consider doing a more detailed user study in the future.

5.6 Discussion

The visualization in our system can be seen as similar to *parallel coordinates* [83], which is a well-known technique for visualizing high-dimensional data. In parallel coordinates, each attribute from the dataset is assigned an axis (coordinate), on which data points are plotted and connected for each record, creating *polylines*. Although several attempts for visualizing nominal data exist [19, 143], parallel coordinates are mostly used to visualize high-dimensional numerical data. The axes are scaled to the minimum and maximum values for each attribute in order to make detection of patterns easier. The visual similarity between the two approaches exists in terms of using equidistant parallel axes as anchors for data objects, however, there are several key differences coming from the different tasks that are performed and the nature of the data that is being visualized.

Parallel coordinates are usually applied on high-dimensional (usually numerical) datasets, where missing values are rare and the goal is to find patterns of similar polylines from different records. In our case, news stories can last for only one day, or they can span over several days, with different start and end dates. Besides, the stories can split and merge, creating a more complex data structure than simple data tuples. Theoretically, we could clone each story cluster that splits into two clusters to create separate data records, but this redundancy does not seem practical and beneficial.

The goal of our visualization is to: 1) identify most important news stories in a longer period of time, and 2) understand their development over time. This differs from pattern detection in parallel coordinates, where the user expects to find similar records across all axes.

Moreover, our "axes" contain ordinal data, i.e. daily story clusters which are sorted by a pre-defined ranking criterion (cluster strength or the number of articles in the cluster). In the interactive exploration phase, the ranking is relaxed by allowing the user to reorder clusters using an algorithm that minimizes edge crossings to improve the legibility of the visualization. This option provides better story tracking and split/merge detection.

Alternatively, we could have used equidistant points on each axis to encode cluster ranking. Since each cluster's size is mapped to its height, this would create gaps between the clusters on each axis and the information about the absolute number of articles for each day would be lost. Our design provides visually more compact representation of the dataset. Finally, the axes in parallel coordinates can be reordered to make comparison between the neighboring axes easier. This possibility, although feasible, is not supported by the real-world tasks in our case, since our attributes (i.e. axes) are days.

5.7 Conclusions and Future Work

In this Chapter, we have presented a visual analytics system for exploration of news stories development and their relationships. Our approach helps in understanding the evolution of long and short stories in a wide time frame, merging and splitting of stories, as well as fine-detailed analysis of story content on three different levels of detail. The incremental processing and visualization of unstructured and semi-structured text data allow the application of the system on the real-world news data streams. The global overview visualization helps in identifying the major news stories over a long period of time and it is enriched with the interaction techniques to filter and re-rank stories on various user-adjustable criteria in order to provide a clutter-free display. Finer-granulated views that correspond to shorter time windows allow analysis of news stories with higher level of detail, up to the textual content of each news article itself. We have demonstrated the effectiveness of our approach on a real-world news stream and described the news stories and their content that were found with our system.

In the future, we plan to refine our document clustering module to enhance the informative context of story labels and test the system with sliding and dynamically adjustable time windows. Additionally, we plan to replace the splitting and merging thresholds, which are currently based on empirically adjusted values to a more refined and less data dependent automatic algorithm. Our research efforts will continue in the direction of integrating incremental text analysis with novel visualization methods that will enable information analysts to analyze and understand growing document collections more effectively and efficiently.

6 Real Time Visual Analytics for Text Streams

Contents

The previous two chapters demonstrated how principles of streaming visual analytics can be used to solve complex analytical tasks using event data from data streams. This chapter explores different experimental methods that combine real time data processing and incremental visualization applied on text streams. First, I present the data acquisition system and the Europe Media Monitor news stream provided by the European Commission's Joint Research Centre. The section is followed by the offline analysis of co-occurrences of entities and entity correlation to illustrate the richness of the data.

The chapter continues with the introduction of a visual analytics algorithm that detects, tracks and visualizes event episodes in news stream in real time with a forgetting model and incremental updates. Next, the *StreamSqueeze* visualization is presented, which is a space-filling technique developed for monitoring event data.

The chapter is concluded with an incremental extension of the CloudLines visualization, which computes densities in partially overlapping time windows and adapts the estimates according to the overlapping region. The technique is applied on customer survey text streams and microblog data to demonstrate its usefulness in other domains.

Parts of this chapter have been previously published in:

Milos Krstajic, Florian Mansmann, Andreas Stoffel, Martin Atkinson, Daniel A. Keim. Processing Online News Streams for Large-Scale Semantic Analysis. ICDE 2010, 1st International Workshop on Data Engineering meets the Semantic Web (DESWeb), 2010. [107][1]

Milos Krstajic, Enrico Bertini, Florian Mansmann and Daniel A. Keim. Visual analysis of news streams with article threads. StreamKDD '10: Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques, ACM KDD 2010, 2010. [105][2]

Florian Mansmann, Milos Krstajic, Fabian Fischer and Enrico Bertini. StreamSqueeze: A Dynamic Stream Visualization for Monitoring of Event Data. SPIE 2012 Conference on Visualization and Data Analysis (VDA '12), 2012. [120][3]

[1]The most of the publication was written by me and supervised, improved and proofread by Florian Mansmann. Andreas Stoffel helped with programming, Martin Atkinson provided the data, while Daniel A. Keim gave research advice.

[2]The idea and the programming were done by me and the paper was jointly written and proofread by all authors.

[3]The idea for the technique and the concept were developed by Florian Mansmann and myself. Florian Mansmann led the writing and implementation and Fabian Fischer helped in implementing. All authors did the cross-checking and proofreading.

6.1 Data Acquisition: Europe Media Monitor News Stream Retrieval

6.1.1 System Architecture

Europe Media Monitor (EMM) [13] is a news aggregator, which collects news articles from over 2,500 sources in 42 languages. These hand-selected sources include media portals, government websites and commercial news agencies. EMM processes 80,000 - 100,000 articles per day, enriching them with various metadata on which we perform our analysis.

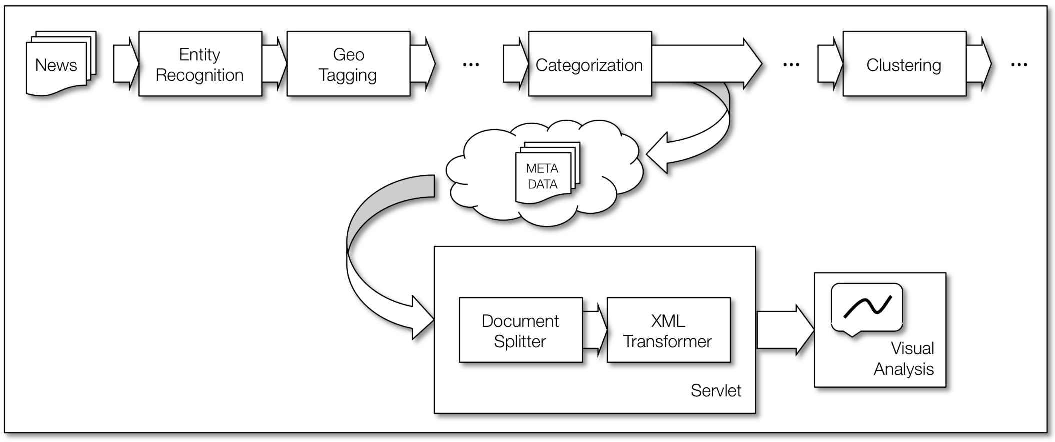


Figure 6.1: News streaming system. The articles are semantically annotated in the pipeline of processing nodes in the EMM system and XML metadata is sent in HTTP posts. On our side, the incoming data is processed within a servlet designed as an external EMM node.

Figure 6.1 presents the overall system architecture. Data retrieval is realized as an extension to the existing EMM web service architecture. Incoming data is handled by a Java servlet, which is responsible for two-way communication with the other side from which it receives HTTP post requests. The servlet is designed as an external EMM processing node connected to the EMM pipeline, whose web service architecture is briefly described in [13].

This push technology allows for immediate retrieval of the new data as soon as it becomes available. Additionally, this architecture also allows us to send the results of our processing back, where it can be included in the

EMM system. Similar nodes exist in the EMM processing chain and each of them performs a specific processing task on incoming data, which is retrieved from the upstream node, and outputs processed data to the next node on the downstream. Queueing and scheduling of jobs is implemented in every node to ensure successful transfers without loss of data. The processing node thereby sends back HTTP response status code 200 to the node on the upstream to acknowledge that the data was successfully received, understood, and accepted. This simple modular architecture allows for easy implementation of new modules (nodes) in the system, or replacement of the old ones.

Incoming HTTP post requests are Unicode XML files containing semantically annotated information in the metadata by modules on the upstream. Very often, several XML documents with news metadata are bundled within one incoming XML file. We perform splitting of the articles into separate files based on the unique article id, which is more suitable for our analysis. The XML metadata is also transformed directly in the servlet to our standardized internal XML format, which is used in our group for the analysis of text.

6.1.2 Event Data Description

Incoming XML file consists of semantically enriched metadata that is of great interest for our analysis, such as entities, categories, geo-tags, URL of the article, source, publishing time, date and language. An example file is shown in Figure 6.2.

People and organizations mentioned in the article are extracted in the entity recognition process [159], which relies heavily on multilingual Named Entity Recognition and Classification (NERC) and cross-lingual information aggregation. They are provided as `<emm:entity>` element, together with additional attributes: *id*, *type*, *count*, *position*, and *name*. Content from the `<emm:entity>` tag and *name* attribute show, respectively, the name of the person or organization, as it appears in the entity database and the name variant. The main reasons for these variations are morphological variants,

```
<item emm:id="bbc-78ee9c05b6c74cf842c409f7c3467c1a">
  <link>http://news.bbc.co.uk/2/hi/asia-pacific/8353451.stm</link>
  <pubDate>2009-11-10T19:19+0100</pubDate>
  <source url="http://news.bbc.co.uk" country="GB" rank="1">bbc</source>
  <iso:language>en</iso:language>
  <category emm:rank="1"
        emm:score="18"
        emm:trigger="warned[2]; deadly[2]; war[1]; fire[2];
        incidents[1]; warns[1]; protesting[1]; nuclear[3];
        warship[1]; weapons[3]; battles[1]; armed[2]; ">
          Security</category>
  <category emm:rank="1"
        emm:score="44"
        emm:trigger="North Korea[5]; Pyongyang[1]; ">
          NorthKorea</category>
  <emm:entity id="1510"
        type="p"
        count="1"
        pos="469"
        name="Barack Obama">Barack Obama</emm:entity>
  <emm:georss name="South Korea"
        id="192"
        lat="37.5424"
        lon="126.935"
        count="1"
        wordpos="23"
        class="0">South Korea</emm:georss>
</item>
```

Figure 6.2: Semantically annotated metadata. Entities, categories and geographic locations extracted from the news articled are passed as content in appropriate elements. Each metadata is enriched with additional information, such as trigger words, scores, type, etc.

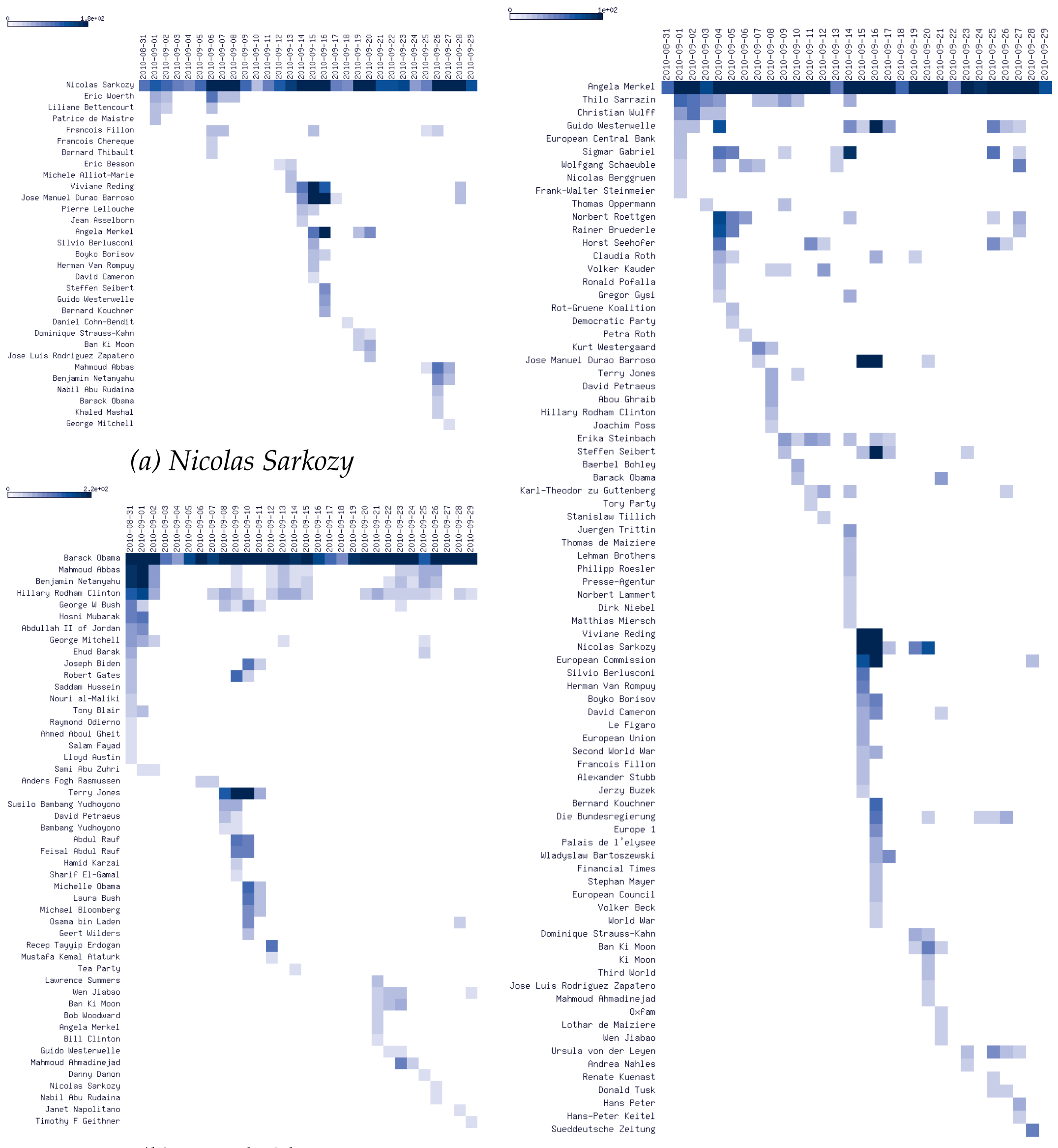
repeated use of the name in the text, spelling mistake or adaptation of the name to local spelling rules. All articles are classified into different categories based on combinations of trigger words which are provided as a value of the `<emm:trigger>` attribute within the `<category>` element. Association of a news article to certain categories gives additional semantic information that can be analyzed further. This is also a logical basis for building a hierarchy over a large collection of news articles in the future. Every article has a `<language>` element, which is added in the language detection process. Semantic analysis of news articles across multiple languages gives the analyst an overview of what is being talked about in different countries. Geoinformation about location mentioned in the articles is provided within `<emm:georss>` element. The method for geolocation recognition and disambiguation in the free text is described in [133].

Incoming data has to be transformed to our internal XML format. Our standardized format is created to streamline structural analysis of any textual data by applying uniform attributes to logical structural elements of text and its meta information. It is used to ensure consistency of work within the group, thus requiring an additional data preprocessing step for the news analysis system.

Therefore, news articles metadata belong to its `<header>` element. Semantic information about entities, countries, categories is converted to `<DOC_ATTR>` elements within the header. Text of the article, which could be retrieved using URL information from the metadata, would belong to `<level>` element within `<body>` of our XML. It should be annotated with attribute *type* with value *section*. Further processing of the text would split the structure of the text into other types, such as *paragraph*, *sentence*, *phrase* or *token*.

6.2 Co-occurrence and Correlation Analysis of Named Entities in Europe Media Monitor Stream

To illustrate the richness of the event data from the stream, I have analyzed co-occurrence and correlation of the named entities, i.e. the names of people and organizations automatically extracted from the news articles . The data was collected in September 2010 and extracted from 257,782 news articles having 53594 unique named entities in 1,036,339 records. The articles are related to two topics: a very general one, *politics*, and a very specific one, *US Open 2010*. This sports event is interesting for analysis since 1) it has a fixed duration of about two weeks; 2) it consists of several short-lived smaller events, such as the 1st, the 2nd round, quarter-finals, semi-finals etc. and 3) the effectiveness of the visualization can be evaluated by comparing the insight gathered in the analysis with daily events that occur during the tournament. Therefore, it can be expected that well-known players who are defeated at a certain stage during the tournament do not appear afterwards in the news reports related to the tournament.



(a) Nicolas Sarkozy

(b) Barack Obama

(c) Angela Merkel

Figure 6.3: Entities co-occurring with Nicolas Sarkozy, Barack Obama and Angela Merkel

6.2.1 Analyzing First Order Co-occurrence Relationships

Examples with entities co-occurring with Barack Obama, Angela Merkel and Nicolas Sarkozy are shown in Figure 6.3 and were taken during September 2010. A first impression is that Angela Merkel has been co-occurring in the news with significantly more people than Barack Obama and Nicolas

Sarkozy. Several patterns stand out. The pattern in the left-central part of Figure 6.3b immediately stands out and it is related to Terry Jones. After reading the corresponding articles, few the days before and around September 11, 2010 the controversial Florida pastor received a lot of media attention for threatening to burn the Quran, which lead to President Obama calling the plan a 'Destructive Act'[4]. The examples in Figures 6.3a and 6.3c show the weakness of the visualization when the underlying data is very sparse. Therefore, the co-occurrence data could be used in combination with CloudLines technique to improve selection of different event types and adjust reordering of event sequences.

Entities in the stream are characterized by different dynamics, where *global* entities that appear throughout the dataset, such as Barack Obama and Angela Merkel, often co-occur with temporal *bursts* of "local" entities, which are usually related to a specific short-lived real-life event. While a streamgraph visualization allows the analyst to spot global and local temporal trends (when there are only few of them in the observed time window), the layout algorithm does not produce good results when the rate of change of co-occurring streams is high in short time intervals. When local maxima overlap in time, these local bursts disturb the layout of other streams and the legibility of the visualization declines.

Such a high rate of change in entity co-occurrence data appears, for example, when global entities are interleaved with local, co-occurring entities. This trade-off between legibility and efficient use of space is highly depending on the underlying data. Therefore, we use a color-coded matrix of entity occurrence over time, where each co-occurring entity is placed in the row of fixed height, which allows for more precise interpretation and gives the analyst a quick overview of single entity first order co-occurrences. Daily frequencies are mapped to color to enable the analyst to identify hot spots easily, without overcrowding the display.

Figure 6.4 gives a quick overview of the tournament with the most prominent named entities appearing in the news for each day. An alternative

[4]http://abcnews.go.com/GMA/president-obama-terry-jones-koran-burning-plan-destructive/story?id=11589122

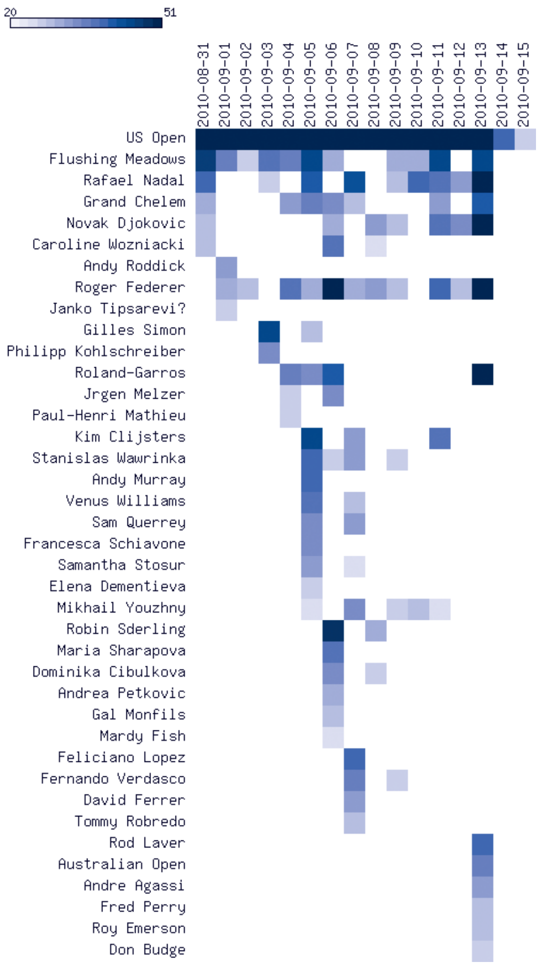


Figure 6.4: Entity co-occurrences in the US Open 2010 dataset

technique is used to show appearance of entities in a matrix representation, where the amount of news articles mentioning a specific entity on a specific date is mapped to color in a corresponding cell. The rows in the cell are sorted by the number of daily occurrences, starting from the earliest date. This simple yet effective visualization allows the user to quickly change the parameters in order to adjust trade off between sparseness of the matrix and level of detail.

In the example shown in Figure 6.4 only entities appearing in more than 25 news articles per day are displayed. It can be easily seen that the most popular players appear until the end of the tournament (Roger Federer, Rafael Nadal and Novak Djokovic, with the latter two playing in the US Open finals), while other players have only temporary peaks. To evaluate the insight that this visualization provides, we have compared the appearance of the players with their performance, results and news reports provided on usopen.org website. Swiss player Stanislas Wawrinka does not appear after September 9, 2010, when he was defeated by Mikhail Youzhny in the men's quarterfinals, who, on the other hand, appears until Sep 11, when he was defeated by Rafael Nadal. Additionally, this player shows an interesting on-off pattern in the period September 5 - September 9, which is appearing because of the tournament schedule. On the day of the tournament finals, a lot of news articles report on the history of US Open, mentioning the Grand Slam winners Rod Laver, Andre Agassi, Fred Perry, Roy Emerson and Don Budge.

6.2.2 Correlation Analysis of Named Entities

To analyze relationships between named entities that co-occur in news articles, we're treating the articles as documents and entities appearing in those documents as terms in order to calculate TF-IDF weights. For the analysis we're using the US Open dataset, which consists of 1103 documents and 1161 terms that gave 13409 TF-IDF values. Since we have the articles from many sources, we can expect to have similar news articles from different news sources that report on the same event. Most of the articles mention only 2-4 players, and these articles are either match reports or other player-centric news (player's comments about the match, opponents, injury, form etc). However, some news articles contain more than 10 entities and, in most cases, these are daily tournament reports. The entities appearing in these documents should be weighted less, since the information they provide is less player-specific. Therefore, we have calculated relative term frequencies of the entities first. Each term in a document that contains 3 terms will have term frequency value tf = 1/3, while in a document containing 10 terms the value will be tf = 1/10. IDF factors are used to lower the weight of terms that occur very frequently in the dataset.

Statistic	TF	IDF	TFIDF
Minimum	0.006	0.301	0.002
Maximum	1	3.0403	1.521
Mean	0.082	1.328	0.099
Std. dev.	0.061	0.716	0.094
Variance	0.004	0.512	0.009
Median	0.071	1.131	0.073
Lower Quartile	0.043	0.74	0.042
Upper Quartile	0.1	1.771	0.122

Table 6.1: US Open entities TF-IDF statistics

TF-IDF weights statistics, shown in Table 6.1 and Figure 6.5, have a lower quartile value of 0.042 and an upper quartile value of 0.122, while maximum value is as high as 1.521. Ranges of outlying values can be seen in the histogram. Instead of creating a correlation matrix of the whole dataset, where outliers would affect the calculation significantly, we have used midspread of TF-IDF values. Additionally, manual inspection of the outliers showed that these documents were not directly related to the US Open tournament, containing entities like Hillary Rodham Clinton, Thilo Sarrazin, Steve Jobs etc, and therefore validated our decision to filter the dataset. After filtering the frequencies in the interquartile range, we were left with 1053 documents and 432 terms.

To calculate linear dependency between recognized named entities, for each term (i.e. named entity) a vector of documents with corresponding

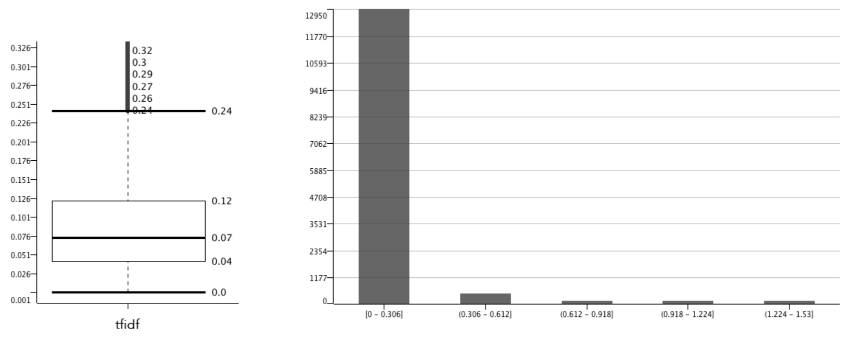


Figure 6.5: US Open entities TF-IDF statistics: boxplot (left) and TF-IDF histogram (right)

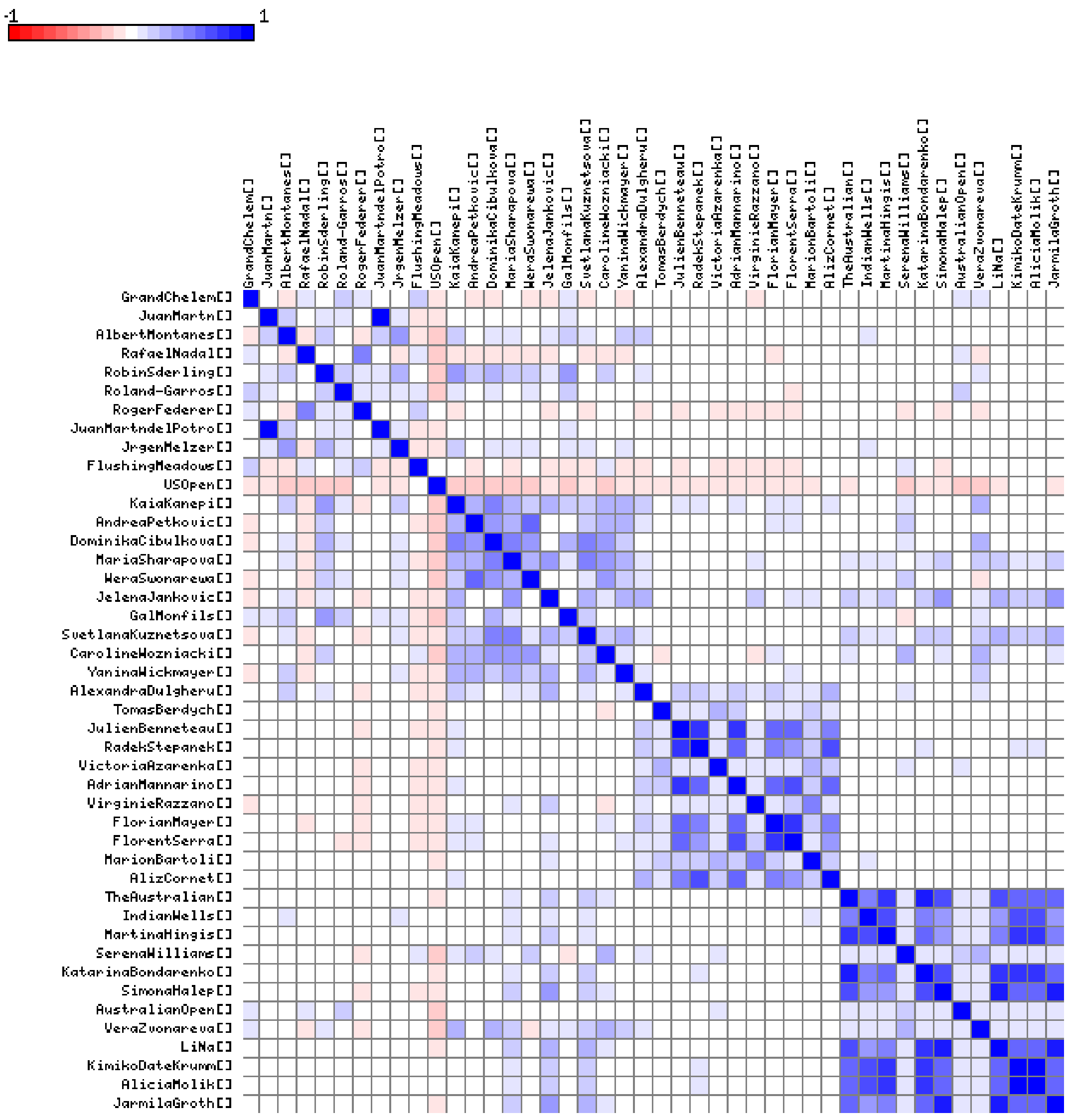


Figure 6.6: Correlation matrix of the named entities in the US Open dataset. A zoomed region with the first 44 entities shows strong correlation within male player clusters and female player clusters. Pearson's product-moment correlation coefficient is calculated for each pair of entities and encoded using a bipolar colormap, from red (-1) through white (0) to blue (1), showing negative, no, or positive correlation, respectively.

TFIDF values is created, and then Pearson's product-moment correlation coefficient is calculated for each pair of terms. We expect that the entities, which appear together in the documents frequently (and therefore are *similar*), to have strong positive correlation, while those that are not linearly dependent would have low or no correlation. The part of the correlation matrix is shown in Figure 6.6 in which positive correlation is visually encoded with different brightness levels of blue, while negative correlation is shown as red, and non-correlated entities are white. The full matrix with all 432 entities is shown in Figure 6.7.

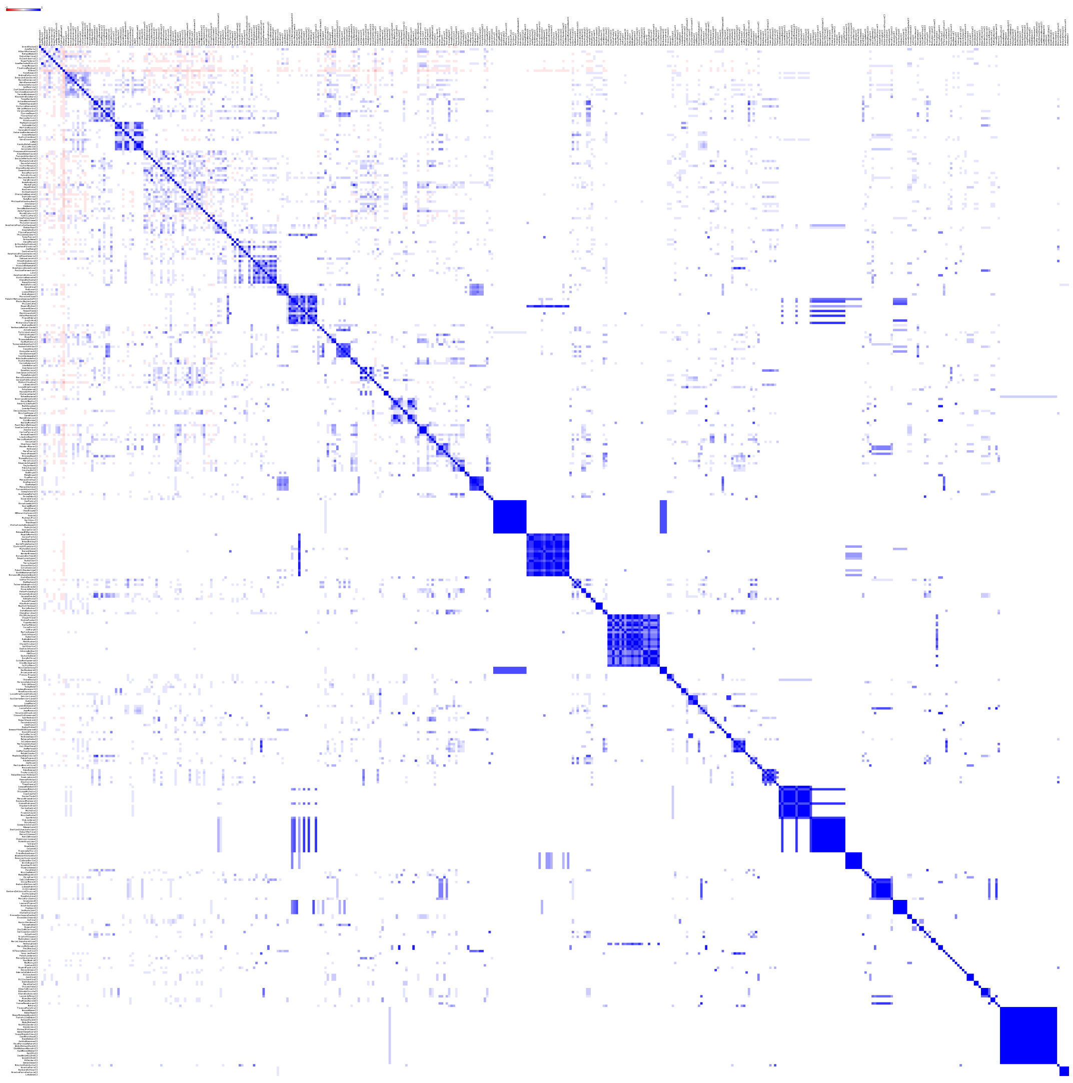


Figure 6.7: The full US Open 2010 entity dataset correlation matrix

6.3 Real-Time Detection, Tracking and Visualization of Event Episodes in News Streams with StreamingThreads

Previous section described the system architecture, data types and the richness of the retrieved data by analyzing co-occurrences and correlation of named entities extracted from a sample dataset. This section focuses on the real time challenges related to analysis of news streams and presents the following visual analytics contributions:

- *Article Thread*, a novel data model for text, which contains age and duration as temporal properties of the data record.
- *StreamingThreads*, a relevance-based streaming visual analytics algorithm for fast detection and tracking of similar documents in a text stream.
- A streaming visualization technique that supports the StreamingThreads algorithm using dynamic ranking.

6.3.1 Introduction

As demonstrated in the previous chapters, performing complex analytical tasks on the streams that contain unstructured and semi-structured text data types is quite challenging. According to design considerations and challenges presented in chapter 2, the data processing modules need to be designed to fit the analytical tasks and to be executed within a tolerable timeframe (*user interaction time*). Existing topic modeling algorithms are too expensive to compute in real-time, yet the data has to be processed in some way to construct meaningful model abstractions for the user. Besides, the visualization needs to be supported by an appropriate updating strategy to inform the user about the changes on time, and to provide the right temporal information about the data.

The age of the events (articles) in the stream has to be taken into account, since the item that was important yesterday might not be important anymore. However, it might not be a good idea to remove all articles from the past, since some of them might still be relevant in the light of current events. Furthermore, keeping all historical data in the visualization is not feasible for both space and performance reasons. This leads to the conclusion that an appropriate abstraction of groups of similar news is needed, which will take into account the special relevance requirements and the age of data. It is necessary to make a distinction between the old news that continuously keep the attention of the media and the latest news on the one side, and the old news that are not relevant anymore on the other side.

There are very few examples in related work that combine both text mining algorithms and visualization in a real time. For example, Rohrdantz et al. presented a visual analysis algorithm to detect sentiment in customer feedback data [141], which has been recently extended into an online event detection algorithm [140] [110].

6.3.2 Article Thread Data Model

Article Thread is a special data model for grouping similar documents, whose purpose is to maintain temporal relevance information of the documents

startDate	Timestamp of the first news item
endDate	Timestamp of the last item in the thread
duration	endDate - startDate
age	time since the item entered the stream
cat	category of the item
tags	list of news article tags
entities	list of entities appearing in the news
connectedFiles	pointers to similar articles
url	URL of the article

Table 6.2: Article Thread Data Model

that are connected to the model. Based on the assumption that the relevance of a document cluster decreases when the cluster is not actively updated, an article thread contains several temporal attributes that keep track of its updating history. The model also contains relevant meta information that is used for fast generation of new clusters. The most important attributes of the model are shown in Table 6.2.

The term *thread* implies similarity to the term *topic* due to its similar purpose and, at the same time, distances itself from it because the approach for creating abstractions from text data is completely different from already established topic modeling methods in the field of text mining.

Article Thread data model has a dual use: first, it is used as a single news data item retrieved from the stream, with common attributes, such as the timestamp and various metadata - url, language and *tags*. The *tags* attribute is represented by a list of categories to which a specific article belongs to, and they are assigned based on the combinations of trigger words that are found in the article. These lists are used to compare the articles and create new threads. Each trigger word has a rank based on the number of appearances of the word in the article and contributes to the final rank of each tag. We use the tag with the highest rank to assign an article to a specific category. An example of the *tags* list is shown in Table 6.3.

Additional *entity* metadata is available in the data stream and can be used in future for finer comparison and aggregation of articles. Entities represent people and organizations mentioned in the news and the details about the named entity recognition process can be found in [159]. An example of entity metadata found in the news is shown in Table 6.4. More details about the specific news data item attributes and the streaming system can be found in [107].

Second, a data object is used in event-based analysis of our stream as the thread originator, through event-specific attributes *endDate*, *duration* and *connectedFiles*.

	tag	rank	score	trigger words
article1	Belgium	1	36	Brussels[2]
				Belgium[1]
				BRUSSELS[1]
	ManMadeDisasters	3	10	accident[1]
				train[1]
				Trains Collide[1]
				collision[1]
	Brussels	2	260	Brussels[2]
				BRUSSELS[1]
	tag	rank	score	trigger words
article2	Iran	3	10	Iran[1]
	Qatar	4	10	Qatar[1]
	Palestine	2	10	Palestinian[1]
	FreedomSecurity	1	29	freedom[5]
				fundamental
				rights[1]
				Freedom[2]
	Society	5	33	women's rights[1]
				freedom of religion[1]
				rights[3]
				Human Rights[1]
	tag	rank	score	trigger words
article3	PropertyCrime	4	50	criminal[1]
	Earthquake	1	52	Haiti[3]
	ElSalvador	2	40	Salvador[2]
				San Salvador[1]
				El Salvador[1]
	Haiti	1	29	Haiti[3]
				Port-au-Prince[1]
	FundamentalRights	1	41	trafficking[5]
				immigration[1]

Table 6.3: Examples of Tag List Attributes

6.3.3 Streaming Visualization of Raw Event Data

A naive approach to visualization of the events in the stream in real-time would be to plot the data point on the screen as soon as it arrives and encode

name	id	type
Saad Eddin Ibrahim	177308	p
Hamad bin Jassim bin Jaber Al Thani	78594	p
Human Rights Council	152035	o
Freedom House	72269	o

Table 6.4: Entity List Attributes

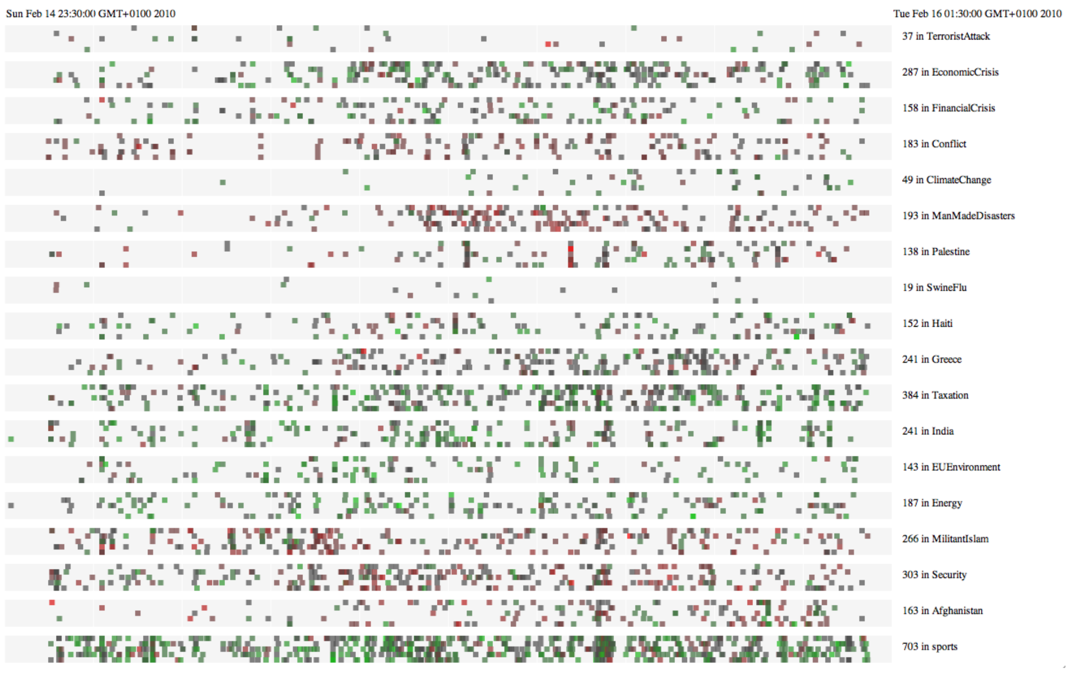


Figure 6.8: Monitoring raw event data. The articles are organized in 18 categories, which are shown on the right with the number of articles in each category. Color is mapped to a tonality value of news article, where saturated green represents a high positive tonality and saturated red represents a very negative tonality. The events in grey are neutral.

the data attributes of interest to different visual features. The visualization of raw event data without aggregation or clustering, as shown in Figure 6.8 is meaningful in monitoring tasks in which:

1. the relationships between the items are not of great importance

2. the size of the time interval in which the monitoring is performed can remain constant.

The snapshot of the data stream was taken during monitoring interval Feb 14, 2010, 23:30 GMT+01:00 - Feb 16, 2010, 01:30 GMT+01:00. Each news item is represented as a 6x6 block of pixels and positioned on the x-axis timeline. The *alpha* value is set to 0.5 in order to show overlapping data points. The color of the item is mapped to its *tonality* score, where saturated green and red represent the items with high positive and negative tonality scores, respectively. Grey color is mapped to neutral articles. This visualization is updated as soon as the new events appear in the stream.

6.3.4 StreamingThreads Algorithm: Detection and Tracking of Event Episodes

I developed StreamingThreads algorithm to dynamically detect event episodes in real time (algorithm 1). The algorithm ranks the episodes based on their relevance and optimizes the performance of the visualization. This data-dependent and time-dependent solution defines relevance based on three important thread characteristics: *age*, *duration*, and *size*.

The algorithm creates and displays recent threads that consist of similar data items, i.e. in case of news stream these are the articles that are reporting on the same event, while keeping the most important (relevant) threads from the past and removing the irrelevant ones. The usual approach is to use *aging*, where each item is removed when its age reaches a specific threshold. In our case, the challenge is that we're not working with data points that are discrete in time, but threads, which are created from discrete data points based on their similarity. Therefore, there should be an obvious difference between two threads, i.e. *relevance*, even if they have the same age. The key idea is to keep relevant events and dismiss the others by taking into account 3 criteria. The thread has to be *old*, *short*, and *sparsely populated* to be deleted.

The algorithm receives two arguments: a, the new item in the stream, and s, the array of threads that have been already created. The algorithm depends on 3 parameters that can be set by the user: the time interval ts, the minimum duration d and the minimum number of items n in a thread e from s.

Removal of Irrelevant Threads

The parameter *ts* is checked against *timeSpan*, which represents the time interval between the timestamp of the new data item *a.startDate* and the timestamp *e.endDate* of the last item in a thread *e*. If *timeSpan* is greater or equal to this threshold, *e* is considered for removal. We consider them less relevant than the recently updated ones. Basically, the parameter *ts* proposes those threads that didn't acquire any new items in a specific period of time as the candidates for removal.

Algorithm 1 StreamingThreads(a:Item,s:Threads)

```
1: set ts, d, n
2: set simThreshold
3: simExists:Boolean for new threads
4:
5: $mainLoop$ :
6: for each $e$ in $s$ do
7:     timeSpan = a.startDate - e.endDate;
8:     if $timeSpan \geq ts$ AND $e.duration \leq d$ AND $e.connectedFiles.length < n$
       then
9:         remove b from s;
10:    else if $a.cat = e.cat$ then
11:        sim = compareTags(a,e)
12:        if $sim > simThreshold$ then
13:            simExists = TRUE;
14:            e.alpha += 0.05;
15:            e.connectedFiles.push(a.lnk);
16:            e.width = a.startDate-e.startDate;
17:            e.endDate = a.startDate;
18:            break $mainLoop$;
19:        end if
20:    end if
21: end for
22:
23: create new thread
24: if $simExists = FALSE$ then
25:     drawItem(a);
26:     s.push(a);
27:     simExists = FALSE;
28: end if
```

The other two parameters *d* and *n* are only thread-dependent, but the conditions are checked in conjunction with the first one. That way the removal procedure is called only when the new data arrives and not in specific time intervals. The attribute *duration* of the thread *e*, from the array *s*, is the time interval between the first (*e.startDate*) and last (*e.endDate*) data item in the thread. If *e.duration* is less or equal to the threshold *d* and the number of items in a thread *e.length* is less or equal to the threshold *n*, the thread *e* becomes a candidate for removal. If all three conditions are met, the thread *e* is removed from the visualization and memory. By adjusting these parameters, we can select different relevance measures for the data we're working on.

Adding a New Article to a Thread

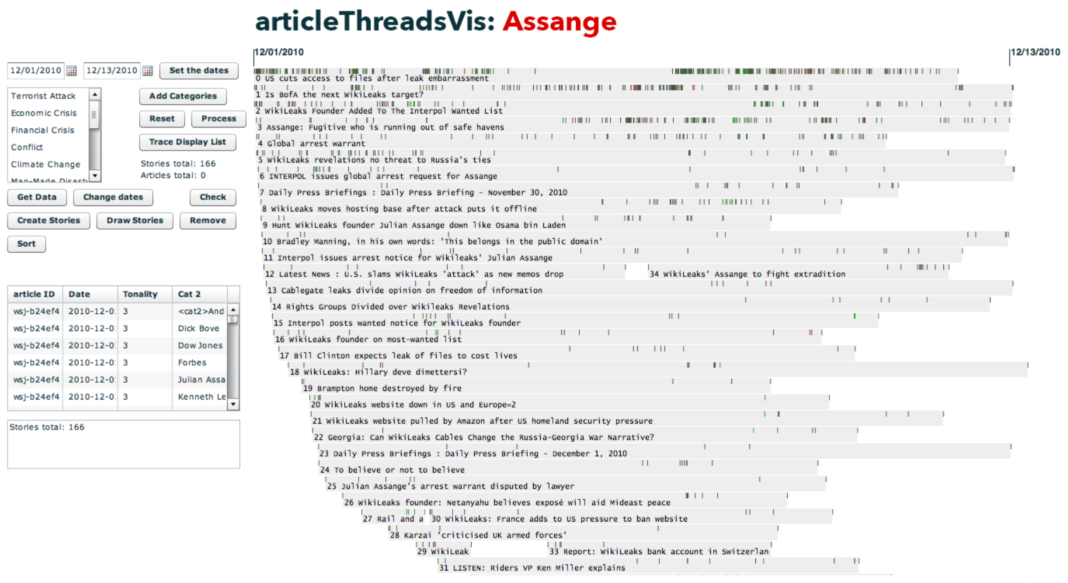
Next, Jaccard similarity coefficient [85] is calculated if the item *a* and the thread *e* are assigned to the same category. If the coefficient is above a certain threshold *simThreshold*, *a* is added to *e* and the visualization gets updated. The color of the event is adjusted to reflect the number of items in it, and its length is extended to include *a.startDate* as the new *e.endDate*. Note that only the pointer to data item *a* and not the whole data object is added to the thread *e* in order to maintain performance of the streaming visualization. At this resolution, we are not interested in any other item detail within a specific thread.

Creating a new thread. Finally, if *a* is not similar to any thread *e* from *s*, *a* is added to *s* as the originator of a new thread.

The algorithm works globally by removing the irrelevant events from all categories when a new event arrives, regardless of its category assignment. It could be easily adjusted to work locally, within a category, if the streaming data would favor this approach.

Visualizing Tracked Article Threads

Figure 6.9 shows the results of the StreamingThreads algorithm related to a story about Julian Assange. The threads are represented as grey horizon-



Stories total: 166
Articles total: 0

article ID	Date	Tonality	Cat 2
wsj-b24ef4	2010-12-0	3	<cat2>And
wsj-b24ef4	2010-12-0	3	Dick Bove
wsj-b24ef4	2010-12-0	3	Dow Jones
wsj-b24ef4	2010-12-0	3	Forbes
wsj-b24ef4	2010-12-0	3	Julian Assa
wsj-b24ef4	2010-12-0	3	Kenneth Le

Stories total: 166

0 US cuts access to files after leak embarrassment
1 Is BofA the next WikiLeaks target?
2 WikiLeaks Founder Added To The Interpol Wanted List
3 Assange: Fugitive who is running out of safe havens
4 Global arrest warrant
5 WikiLeaks revelations no threat to Russia's ties
6 INTERPOL issues global arrest request for Assange
7 Daily Press Briefings : Daily Press Briefing - November 30, 2010
8 WikiLeaks moves hosting base after attack puts it offline
9 Hunt WikiLeaks founder Julian Assange down like Osama bin Laden
10 Bradley Manning, in his own words: 'This belongs in the public domain'
11 Interpol issues arrest notice for WikiLeaks' Julian Assange
12 Latest News : U.S. slams WikiLeaks 'attack' as new memos drop
34 WikiLeaks' Assange to fight extradition
13 Cablegate leaks divide opinion on freedom of information
14 Rights Groups Divided over WikiLeaks Revelations
15 Interpol posts wanted notice for WikiLeaks founder
16 WikiLeaks founder on most-wanted list
17 Bill Clinton expects leak of files to cost lives
18 WikiLeaks: Hillary deve dimettersi?
19 Brampton home destroyed by fire
20 WikiLeaks website down in US and Europe=2
21 WikiLeaks website pulled by Amazon after US homeland security pressure
22 Georgia: Can WikiLeaks Cables Change the Russia-Georgia War Narrative?
23 Daily Press Briefings : Daily Press Briefing - December 1, 2010
24 To believe or not to believe
25 Julian Assange's arrest warrant disputed by lawyer
26 WikiLeaks founder: Netanyahu believes exposé will aid Mideast peace
27 Rail and a
30 WikiLeaks: France adds to US pressure to ban website
28 Karzai 'criticised UK armed forces'
29 WikiLeak
33 Report: WikiLeaks bank account in Switzerlan
31 LISTEN: Riders VP Ken Miller explains

Figure 6.9: Visualization of the StreamingThreads algorithm results.

tal bars with the startDate and the current endDate attributes determining their length. Tick marks represent articles within a thread and their color is mapped to the tonality of the article. The title of a representative article is given within each bar. The threads are dynamically stacked using ranking criteria similar to priority queues, e.g. threads 33 and 34 are automatically added to the empty space since threads 12 and 29 have ended. The algorithm takes the following parameters: ts = 240 (min), d = 200 (min) and n = 10.

6.3.5 Discussion

Visualization of events without dynamic clustering (Figure 6.8) provides direct access to exploration of single items, but lacks information about relationships between the items. Also, it has the inability to make a difference between important and unimportant items, except for sudden bursts within a category. However, a sharp increase in the number of news articles that belong to the same category in short time interval doesn't necessary have to mean that these articles are related to the same event. This change could be also caused by the fact that a specific source published different news articles in intervals that are very close to each other. The aforementioned issue can be easily visible in very generalized categories with a lot of data, such as *sports*.

Second, visualization is non-discriminative, which means that all the items that exist on the screen and in the memory are of equal importance. In many applications, removal of irrelevant items is needed, both for the analytical and performance issues.

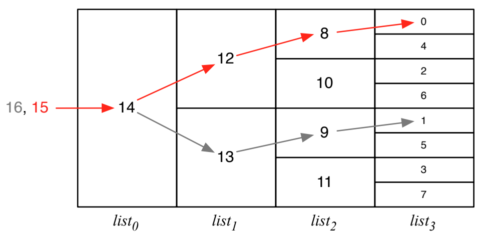
6.4 Real Time Monitoring of Event Streams with StreamSqueeze

This section introduces an experimental technique for visualization of events in streaming text data. It is a hybrid space-filling approach that combines static item positioning for better readability of text labels and smooth transitioning of items to facilitate their traceability when making space for new and updated information. The technique builds upon two basic conditions:

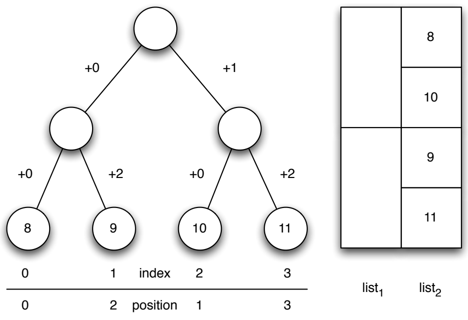
- *C1*: New events in the data stream have higher relevance due to their more recent nature.
- *C2*: Smooth transitions enable traceability of items on the screen.

By taking the first condition into account, the proposed layout algorithm arranges items in several lists of various size on the screen and optimizes the positions within each list so that the transition of an item from one list to the other triggers least visual changes. As a consequence of the second condition, our animation scheme ensures that during a screen update 50 percent of items in each list stay at static positions where reading is most effective while the remaining items continuously shrink and move to the adjacent screen position in the subsequent list.

We show that our approach has several favorable properties: 1) recent items appear larger and can thus be shown with more details, 2) scalability



(a) Update schema of StreamSqueeze: When a new item is added to $list_0$, the oldest item with smallest index of this list is moved to the next list. Each of the moved items make space for the one from the previous list.



(b) Item positioning scheme shown for $list_2$ with a binary tree: Given the alternating updates of items in $list_1$, the changes in $list_2$ should occur in the same row under the condition that after four updates every item in $list_2$ has been updated.

Figure 6.10: Update and positioning scheme of the StreamSqueeze *visualization.*

due to the screen-filling nature of the approach and 3) local changes and continuous animation allow tracking of items. Besides showing these properties, we demonstrate the usage of our technique on large real-world streams in the domains of news and system log monitoring.

The technique is shown in Figure 6.10a. The events are ordered in vertical lists, with the latest events being the largest (condition C1). Each subsequent list then contains twice as many items, which are scaled down accordingly.

Updating the visualization is a key aspect for analysis of events in dynamic information streams. For this particular visualization, this means that, as soon as a new item comes in, we remove the oldest one of the list, place it in the next column, and repeat this procedure over all columns shown on the screen as shown by the red arrows in Figure 6.10a. The red arrows show one update sequence replacing $item_{14}$ with a new one, $item_{12}$ with $item_{14}$, $item_8$ with $item_{12}$ and $item_0$ with $item_8$. Note that the gray arrows show the update sequence for the next update.

6.4.1 Layout Function

Given the *StreamSqueeze* update schema, the horizontal positioning, or in other words the assignment of each item to a list, is trivial to compute based on the age order of the events, in which $a = 0$ marks the latest event and $a = 2^{|lists|} - 2$ the oldest displayed event:

$$list(a) = \lceil log_2(a + 1)\rceil \tag{6.1}$$

As an example, after item 12 is replaced by 14 as shown in Figure 6.10a, its new age is $a = 3$ since there are three more recent events currently displayed and it is thus assigned to $list_2$.

Since we aim for smooth transitions rather than abrupt positional changes to enable traceability of items (C2) when moving an item from one list to the next one, we need to ensure that the item's positions before and after its update are adjacent. To achieve this, the vertical alignment function for placing the items within each list is based on a lookup of the index value modulo the size of the target list in a binary tree containing all values from 0 to 2^i, where i is the list id. Each time the binary tree is traversed to the right, we add 2^l, where l is the next level of the tree, as illustrated in Figure 6.10b. The array (0,2,1,3) determines the positions of items 8 to 11 as shown in Figure 6.10b. When the new item 12 arrives we need to remove item 8.

Given the above presented horizontal and vertical placement schemes, we can guarantee that after a new item is added, the position of an item either stays the same or it is moved to an adjacent screen position in the subsequent list. In the latter case, the moved item is scaled down by a factor of two. Since only the screen position of one item per list is changed, we can guarantee that this visualization technique triggers least visual changes.

Note that for using the *StreamSqueeze* technique with high-frequency streams in practice, the first few lists are often discarded as shown, for example, in Figure 6.11. The benefit of this is that the animated screen becomes more balanced in the sense that not every update completely changes the first list or half of the second one.

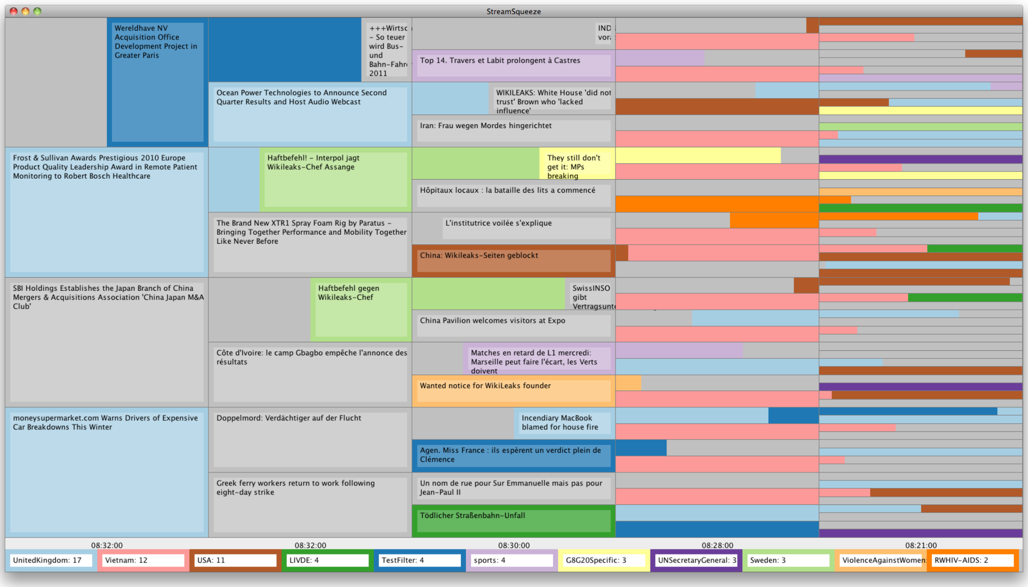

StreamSqueeze
Wereldhave NV Acquisition Office Development Project in Greater Paris
+++Wirtsc – So teuer wird Bus– und Bahn–Fahr 2011
IND vor
Top 14. Travers et Labit prolongent à Castres
Ocean Power Technologies to Announce Second Quarter Results and Host Audio Webcast
WIKILEAKS: White House 'did not trust' Brown who 'lacked influence'
Iran: Frau wegen Mordes hingerichtet
Frost & Sullivan Awards Prestigious 2010 Europe Product Quality Leadership Award in Remote Patient Monitoring to Robert Bosch Healthcare
Haftbefehl! – Interpol jagt Wikileaks–Chef Assange
They still don't get it: MPs breaking
Hôpitaux locaux : la bataille des lits a commencé
The Brand New XTR1 Spray Foam Rig by Paratus – Bringing Together Performance and Mobility Together Like Never Before
L'institutrice voilée s'explique
China: Wikileaks–Seiten geblockt
SBI Holdings Establishes the Japan Branch of China Mergers & Acquisitions Association 'China Japan M&A Club'
Haftbefehl gegen Wikileaks–Chef
SwissINSO gibt Vertragsunt
China Pavilion welcomes visitors at Expo
Côte d'Ivoire: le camp Gbagbo empêche l'annonce des résultats
Matches en retard de L1 mercredi: Marseille peut faire l'écart, les Verts doivent
Wanted notice for WikiLeaks founder
moneysupermarket.com Warns Drivers of Expensive Car Breakdowns This Winter
Doppelmord: Verdächtiger auf der Flucht
Incendiary MacBook blamed for house fire
Agen. Miss France : ils espèrent un verdict plein de Clémence
Greek ferry workers return to work following eight–day strike
Un nom de rue pour Sur Emmanuelle mais pas pour Jean–Paul II
Tödlicher Straßenbahn–Unfall
08:32:00
08:32:00
08:30:00
08:28:00
08:21:00
UnitedKingdom: 17
Vietnam: 12
USA: 11
LIVDE: 4
TestFilter: 4
sports: 4
G8G20Specific: 3
UNSecretaryGeneral: 3
Sweden: 3
ViolenceAgainstWomen
RWHIV–AIDS: 2

Figure 6.11: StreamSqueeze is a visualization technique for dynamic event data that 1) assigns more screen space to the latest events, 2) allows for better readability due to partially fixed text positions and 3) fosters traceability through animation and predictable positions (the events move from the leftmost column to right).

6.4.2 Animation Concept and Age Hints

As stated in condition C2 in the introduction, we assume that traceability can be achieved by smoothly moving items from one list to the next. However, in order to do this, we need to sacrifice the static positions of the items within a list at least to a certain degree.

In the first half of their lifetime within a column, the items' positions stay constant on the screen for better readability and easier interaction. The animation concept starts moving items out of the list once 50 percent of the list's items are added more recently than the considered item. The size of the item in the old list thus gives the user an indication of its forthcoming movement with the next update. Note that setting another percentage is possible, but low values result in fast movements that unnecessarily attract user attention and high values compromise the static text positioning.

Figure 6.11 illustrates how, for example the light green items in the center are smoothly moved out of the 2nd list into the third one. Note that the animation of the brown item in the center into the right 4th list starts earlier than its movement out of the 3rd list. This is due to the fact that items in this list move only half as fast as in the middle one. Since the last list contains more items than all previous columns together and since we only want to show one item at a continuous screen position, we refrain from animating all items of a column at once. Note that the items of one list contain several timestamps, there are several options of displaying the covered time range underneath each column, such as the first, last or median timestamp of the items. For the above referenced figure, we decided that the median is a good representation since it reduces the likelihood of identical timestamps next to each other when displaying labels of the full range. However, depending on the application scenario, the timestamp of the first or last item or the covered time could also reveal important real-time aspects of the data to the analyst.

Since many of the news headlines had identical timestamps, we added some time base jitter to make changes more traceable rather than adding many items at once. For demonstration purposes, we also artificially increased the speed of the news.

Variations of StreamSqueeze

While we have shown that the proposed layout scheme introduces least visual changes under the given assumptions, this scheme completely blocks the visual position variables for other purposes. For many temporal visualizations, however, the x position is used to express the detailed temporal aspects of the shown items. While this can create overlap, it enables analysis tasks that are linked to interval-scaled aspects of time, which is not possible in our proposed layout. As an alternative layout, we therefore offer the option to sort certain lists of StreamSqueeze according to the temporal order of its contained items, which turns the temporal aspects into an ordered scale. As an example, this could be useful when monitoring stock trades since the temporal order of trades much defines future stock prices. A more complex option would be to introduce empty items into the screen to better express interval-scaled aspects of the sorted lists, which we will not investigate in this paper.

Analog to this temporal ordering, we also offer sorting based on metadata attributes, such as news categories or server names in the application examples in Section 6.4.5. This has the advantage, that it is possible to estimate the number of items in each category over the stream's history shown on the screen. Even with small item sizes, the combination of coloring and ordering can foster such insights.

In general, however, a major drawback of such sorted lists is that this destroys both the guarantee of least visual changes under the assumptions as well as the animation concept since it is likely to happen that an item from one list is moved to a non-adjacent screen position in the subsequent list. Note that the latter is even the case when two subsequent lists are sorted according to metadata categories since it is not guaranteed that the proportions of items of the lists stay constant over time. Since it is therefore not possible any more to animated items from one list into the other in the sorted layout, we use a light version of the item's color to give a hint of age and movement behavior to the users as shown in Figure 6.12.

6.4.3 Coloring Strategies

Coloring is a very powerful way to visually group items, which semantically belong together. In our concrete case, we color news items with identical categories or syslog messages from identical servers with the same color. In these cases, we restrict ourselves to 12 qualitative colors as suggested by the online application ColorBrewer [76]. Depending on the requirements of the interaction, red is reserved for marking moving items on the screen.

If the number of categories exceeds this threshold of twelve, we need to overcome a technical and perceptual issue. The simplest solution would be to reuse colors, which can result in misinterpretation. Note that the sorted layout variation is more suitable for such reassignments since sorting the categories for the layout reduces the likelihood that identically colored categories end up next to each other in the layout. This would only be the case if the eleven categories between them in the sorting order are missing in that particular list.

Our current solution overcomes the problem in a different manner by keeping counters for the number of items in each category and assigning a unique color to the twelve biggest categories. Items of all other categories will be drawn with gray background.

A drawback of this approach can be that there will be alternating assignments of gray and the least frequent color to two different categories resulting in a distracting unstable coloring scheme. However, this can be overcome by setting a threshold for reassigning the twelve colors, which is a value above the number of items in the least frequent colored class. For example, the least frequent colored category "politics" has 27 items, but will not be replaced by the still gray category "health" with 28 items since the threshold value is set to 33. Furthermore, we protocol the minimum age for each coloring class so that we can reduce the risk of alternating color assignments for the small classes.

6.4.4 Interaction Concept

StreamSqueeze is meant as an interactive visualization technique, in which the user can mark certain items in the screen, pause and resume the stream. While clicking moving objects is a challenge in itself, our proposed visualization technique has the favorable property that larger items in the earlier lists, which are essentially easier to click move faster than smaller items, which are harder to mark. This counterbalances the difficulties of user interaction with moving objects at least to some degree.

Another usage of the stream visualization is pausing and resuming of the streaming visualization in order to investigate certain items in detail. While pausing is easy to implement, resuming becomes a lot more difficult since the user often needs to understand what has happened in the meantime. Our current solution to this is that after resuming we speed up the paused items with a user defined constant until the real-time end of the stream is reached.

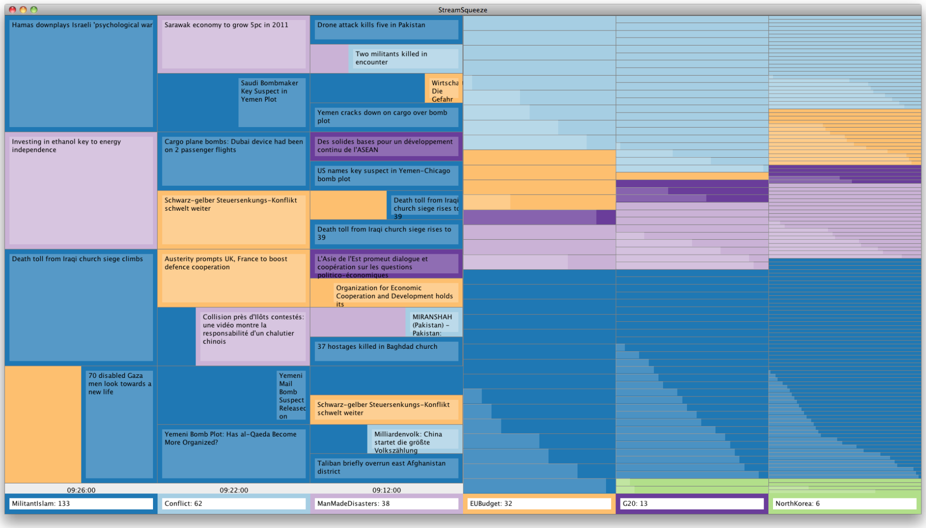

StreamSqueeze
Hamas downplays Israeli 'psychological war
Sarawak economy to grow 5pc in 2011
Drone attack kills five in Pakistan
Two militants killed in encounter
Saudi Bombmaker Key Suspect in Yemen Plot
Wirtscha Die Gefahr
Yemen cracks down on cargo over bomb plot
Investing in ethanol key to energy independence
Cargo plane bombs: Dubai device had been on 2 passenger flights
Des solides bases pour un développement continu de l'ASEAN
US names key suspect in Yemen-Chicago bomb plot
Schwarz-gelber Steuersenkungs-Konflikt schwelt weiter
Death toll from Iraqi church siege rises to 39
Death toll from Iraqi church siege rises to 39
Death toll from Iraqi church siege climbs
Austerity prompts UK, France to boost defence cooperation
L'Asie de l'Est promeut dialogue et coopération sur les questions politico-économiques
Organization for Economic Cooperation and Development holds its
Collision près d'Ilôts contestés: une vidéo montre la responsabilité d'un chalutier chinois
MIRANSHAH (Pakistan) - Pakistan:
37 hostages killed in Baghdad church
70 disabled Gaza men look towards a new life
Yemeni Mail Bomb Suspect Releasec on
Schwarz-gelber Steuersenkungs-Konflikt schwelt weiter
Yemeni Bomb Plot: Has al-Qaeda Become More Organized?
Milliardenvolk: China startet die größte Volkszählung
Taliban briefly overrun east Afghanistan district
09:26:00
09:22:00
09:12:00
MilitantIslam: 133
Conflict: 62
ManMadeDisasters: 38
EUBudget: 32
G20: 13
NorthKorea: 6

Figure 6.12: StreamSqueeze with sorting of the last three lists applied for crisis news monitoring. The six news categories are: North Korea, Militant Islam, G20, Man-made Disasters, Conflict, and EU Budget. The legend for color mapping is shown in the bottom row.

6.4.5 Case Study

The performance of the presented data streaming visualization technique in real-world environment depends on three important factors: First, the frequency of incoming events, second the number of monitored categories, and third the level of detail for each event.

There is an obvious trade-off that has to be made between these factors, which depends on the application and the task. For example, if the frequency of incoming event data is very high, the level of detail that can be comprehended by the user in a monitoring scenario will be quite low. However, if it is necessary to show the content of the latest events in higher detail, the data stream might have to be filtered and/or throttled down to accommodate the user preferences. Additionally, the performance of the technique depends on external factors, such as available screen size and resolution.

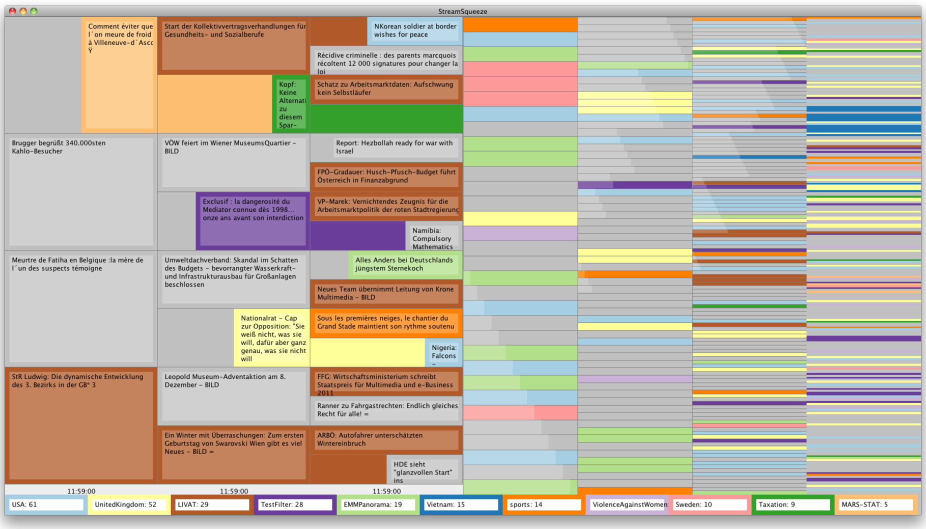
We have deployed the algorithm to work with the news streaming system that was described in section 6.1 in two different monitoring scenarios. In our first case, we have selected six news categories that would be considered important for the analyst in a news monitoring scenario: Man-made

Disasters, North Korea, Conflict, EU Budget, Militant Islam, and G20. The user should be aware of any abrupt changes that could arise in real-time, and in this case the tool serves as an alerting system that can offer focus on the latest events, and provides additional details on demand. At the same time, these rapid changes that occur in the present have to be put into context of the development of events in this and all the other categories of interest in the past. Figure 6.12 shows a snapshot of the tool in which we are using the sorted variation of StreamSqueeze for the leftmost three lists. In this case, the items are sorted according to their news category, which is encoded by color.

The news data items are enriched with various meta-information that should be available to the user, either immediately or on demand. In order to allow the user to observe and interact with any of the items in the stream, the change of item's position should be kept minimal, which is in concordance with our algorithm's criterion of least visual changes. Additionally, the user can decide if the first columns, which can contain only 1, 2, or 4 events, should be kept or discarded from the visualization. Since the recent items appear larger than the old ones, they can be shown in more detail for quick inspection. Depending on the display size, the resolution, and the number of events in a single column, we can choose which metadata should be provided in the available item design space. We use the headline, but we could also use the tags that describe the article, or the named entities appearing in the news. An alternative would be to render the text in these lists significantly larger in relation to the available space. As an article shifts into the past, we limit the amount of information that can be presented in the display object, and we can use only the available named entities that are mentioned in the article, so that the user can get a brief idea about the individual news content.

Since the news aggregator gathers data from a large number of news websites, very often these different sources publish news articles that report on the same real-world event. The bigger the event, the more sources will report on it in a shorter period of time, and visual clustering occurs, where items of the same category next to each other are related (*locality* of events).

The drawback of our layout sorting technique and the visual clustering it provides is that it strongly depends on the generality of the category.



Comment éviter que l´on meure de froid à Villeneuve-d´Ascq Ÿ

Start der Kollektivvertragsverhandlungen für Gesundheits- und Sozialberufe

NKorean soldier at border wishes for peace

Récidive criminelle : des parents marcquois récoltent 12 000 signatures pour changer la loi

Kopf: Keine Alternati zu diesem Spar-

Schatz zu Arbeitsmarktdaten: Aufschwung kein Selbstläufer

Brugger begrüßt 340.000sten Kahlo-Besucher

VÖW feiert im Wiener MuseumsQuartier – BILD

Report: Hezbollah ready for war with Israel

FPÖ-Gradauer: Husch-Pfusch-Budget führt Österreich in Finanzabgrund

Exclusif : la dangerosité du Mediator connue dès 1998... onze ans avant son interdiction

VP-Marek: Vernichtendes Zeugnis für die Arbeitsmarktpolitik der roten Stadtregierung

Namibia: Compulsory Mathematics

Meurtre de Fatiha en Belgique :la mère de l´un des suspects témoigne

Umweltdachverband: Skandal im Schatten des Budgets – bevorrangter Wasserkraft- und Infrastrukturausbau für Großanlagen beschlossen

Alles Anders bei Deutschlands jüngstem Sternekoch

Neues Team übernimmt Leitung von Krone Multimedia – BILD

Nationalrat – Cap zur Opposition: "Sie weiß nicht, was sie will, dafür aber ganz genau, was sie nicht will

Sous les premières neiges, le chantier du Grand Stade maintient son rythme soutenu

Nigeria: Falcons –

StR Ludwig: Die dynamische Entwicklung des 3. Bezirks in der GB* 3

Leopold Museum-Adventaktion am 8. Dezember – BILD

FFG: Wirtschaftsministerium schreibt Staatspreis für Multimedia und e-Business 2011

Ranner zu Fahrgastrechten: Endlich gleiches Recht für alle! =

Ein Winter mit Überraschungen: Zum ersten Geburtstag von Swarovski Wien gibt es viel Neues – BILD =

ARBÖ: Autofahrer unterschätzten Wintereinbruch

HDE sieht "glanzvollen Start" ins

Figure 6.13: StreamSqueeze applied on top 11 news categories. Less frequent news categories were colored in gray. Temporal sorting of the last four lists reveals information about the temporal extent of news topics as seen on the compact dark blue topic in the "Vietnam" category.

In our second example (Figure 6.13) we do not filter the stream or restrict ourselves to the six pre-selected categories. The tool is set to monitor eleven categories that are most frequent in the stream. The advantage of this approach is that these most frequent categories will always stand out, and if the breaking event happens, we would immediately be able to see the context of the event within the category and its recent past behavior. Similarly to the first example, the category labels are placed in the bottom of the screen. As shown in the figure, the temporal sorting indicates if it is continuously reported on certain categories or if bursty events with a compact temporal extent occur such as the dark blue pattern in the last list indicating a story that was simultaneously reported on in the "Vietnam" category. Note that the imperfections of the sorting as shown by the step patterns in lighter colors relate to identical timestamps. In such cases, jitter is added for making the animation smoother, but the original timestamps are kept for sorting the data.

6.4.6 Discussion

Assessment of StreamSqueeze

The first and obvious advantage of the StreamSqueeze technique is that space is made for new information to be displayed in a dynamic usage scenarios. In particular, recent items always appear at the same column, in which the items can be depicted larger and with more details.

Second, while other streaming techniques render items with either constant size or employ aggregation concepts, the screen-filling nature and continuous adaptation of the shown details according to the items' relative age make the technique scalable. In the example shown in this paper the number of displayed items ranges from 124 to 508. However, the visualization scheme can be extended in a straight forward way to show 2047 items (1024 items of 1 pixel height in the last column). Using an adapted version with several columns containing an equal number of items at the end can push this limit even further.

Third, local changes and continuous animation allow tracking of items on the screen across subsequent lists. Since the original layout of StreamSqueeze always moves an item into one of the two adjacent positions in the next list, we can guarantee that an update only triggers positional changes of one item per list and that these changes are all grouped in the horizontal row starting at the largest item in the first list and ranging to the last list. Therefore, we can claim that when discarding the animation concept, our visualization technique triggers least visual changes to the layout, which are horizontally grouped rather than scattered over the whole screen. Of course, this only holds under the assumption that we need to shift one item per list with each update.

While the animation concept of StreamSqueeze relaxes the condition of least visual changes to the screen, but give more support to enable traceability of items across the lists by continuously moving items.

One drawback of StreamSqueeze is that changing size, proportions and sometimes even removing text labels of the items when moving them from one list to the other do not allow for easy recognition of items. However, the screen position of each item canonically defines its previous positions in the lists with less items. Furthermore, interactive hints when hovering the mouse over an item without a text label shows more details and keeping a consistent color coding of the items supports recognition of an item to some degree.

In contrast to classical temporal visualization techniques, StreamSqueeze boldly substitutes the interval-scaled aspect of each item by a rough binning based on the logarithmically-scaled temporal order of the streamed events. For each such bin, represented by a list on the screen, only one timestamp is shown. However, visual hints such as partly moved items or items partly drawn with a white overlay compensate for this rigorous design decision.

Assessment of Variations, Coloring and Interaction

Optimizing the default layout scheme for special purposes therefore triggers many changes to the overall visualization concept. However, some usage scenarios significantly benefit from the sorting options, especially in the columns representing older events. Since the items of these columns are rendered so small that we cannot show text labels any more, we have less motivation to make them stay at constant screen position throughout their lifetime within that column. Major benefits of such a sorting on either the temporal aspects or the metadata assigned to these columns enables analysis tasks related to pattern detection in the frequency or the sequence of events through their colored representations.

However, a sorting destroys the original animation concept that draws part of the older items of a list in their adjacent future screen position within the next list. In such cases, an opaque white overlay indicates when the item is going to be moved out of the list.

Since coloring is an important aspect for recognition of items and for their visual grouping, we defined a constant color for each event category. Since the used 12 color quantitative scale from ColorBrewer [76] is quite limiting, we decided to draw the remaining categories in gray. In many cases, the red color from this scale is reserved for interactive marking purposes.

In general, interaction with moving items on a high-frequency screen is quite challenging. Due to the fact that largely drawn recent items change their screen positions quicker than older ones, the increase in size makes these fast moving items easier to mark. In contrast, slowly moving items are drawn considerably smaller in size.

6.4.7 Conclusions

The presented streaming visualization technique uses a space-filling layout to visualize information streams by taking into account the higher relevance of recent events and the need for making individual items traceable. The main benefits our our technique are:

1. More screen space to the latest events due to their increased relevance for monitoring tasks.
2. Partially fixed text positions on the screen results in better readability of event labels.
3. Our technique triggers least visual changes in the event lists and only localized changes on the screen to foster traceability in an animated real-time monitor.

The first case study detailed how the static sorted layout can be used to track topics in a large-scale online news stream. In this scenario, it was very important to keep fast updated items in the first few columns at static positions until they move to the columns on the right. As soon as labels became unreadable grouping of the prominent categories was more important than maintaining their static positions.

6.5 Real-time Analytics of Event Episodes with Incremental CloudLines

This section describes the incremental extension of the CloudLines visualization, which is applied in two new domains: visual analyses of a customer feedback stream with approximately 50,000 surveys and a microblogging stream from VAST Challenge 2011[5] with more than 1,000,000 messages. The disadvantage of the original technique, which was introduced in Chapter 4 is its density estimation algorithm. It relies on estimating densities from the global distribution in event sequences, and, therefore, adding new events requires either recalculation of all estimates or an intelligent way to combine the estimates from successive time intervals. The incremental extension integrates a variant of the event detection algorithm presented in [140]. Additionally, the examples demonstrate how the technique can be used to validate the event detection algorithm and help the researcher to improve it.

The goal of event episodes detection in customer feedback streams is to identify critical issues related to products mentioned in the survey. The automatic analysis should save time for the customer support teams who would otherwise have to read the surveys manually. Event episodes are detected in event sequences of selected keywords, which suddenly start appearing unexpectedly often in the stream. After detecting the episodes, a human analyst has to explore, interpret and understand these episodes, which involves accessing and reading documents, and putting each event episode into a temporal context. Consequently, it is necessary to provide the analyst with a visual display where s/he is able to perform her/his exploration tasks.

6.5.1 Introducing Incremental CloudLines

The CloudLines technique shows long temporal event sequences in limited space, where the density distribution of the events determines the shape of the sequences. Since the data is not aggregated beforehand, each event on

[5]http://hcil.cs.umd.edu/localphp/hcil/vast11/

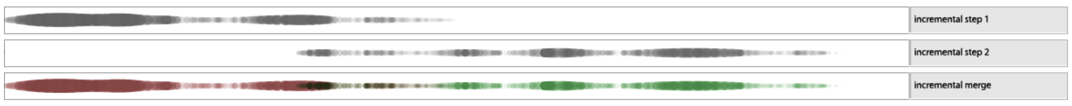


Figure 6.14: Creation of the incremental CloudLine. The density estimates are calculated and adjusted in overlapping sliding windows (top and middle rows), and the merged result (bottom) shows the differences between the first (red) and the second (green) step.

the timeline can be accessed using lens magnification techniques. Applying the technique independently in successive time windows would create abrupt changes between the windows due to the kernel density estimation method, since the kernel function has to satisfy the following condition:

$$\int_{-\infty}^{\infty} K(x)\,\mathrm{d}x = 1$$

Depending on the time window selection and the amount of data in the window, the condition that the integral of the kernel function is equal to 1 would mean that the estimates in the windows of same sizes, but different data volumes, could have values that can even have different orders of magnitude. In order to design a better solution to this problem, we are using overlapping sliding time windows, where the overlap interval is used as a local correction mechanism. The size of the overlap area can be either data-driven or time-driven. In this case the overlapping area contains 15% of the events from the previous time window. The estimates from both windows are compared and the final estimates are calculated as arithmetic means. Afterwards, the radii of the existing objects, which correspond to the data points in the overlap area are adjusted to the new values using animation. The method is shown in Figure 6.14.

This approach provides smooth incremental updates of the streaming visualization with local adjustments, which is necessary when the volume of the data is high. However, the adjustments increase the error and reduce the

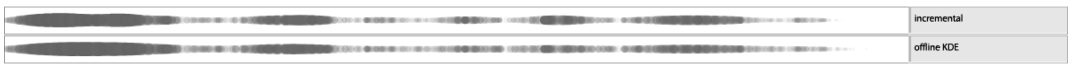


Figure 6.15: Comparison of the incremental and the original CloudLine. The necessary adjustments can be executed when the time and resource constraints allow.

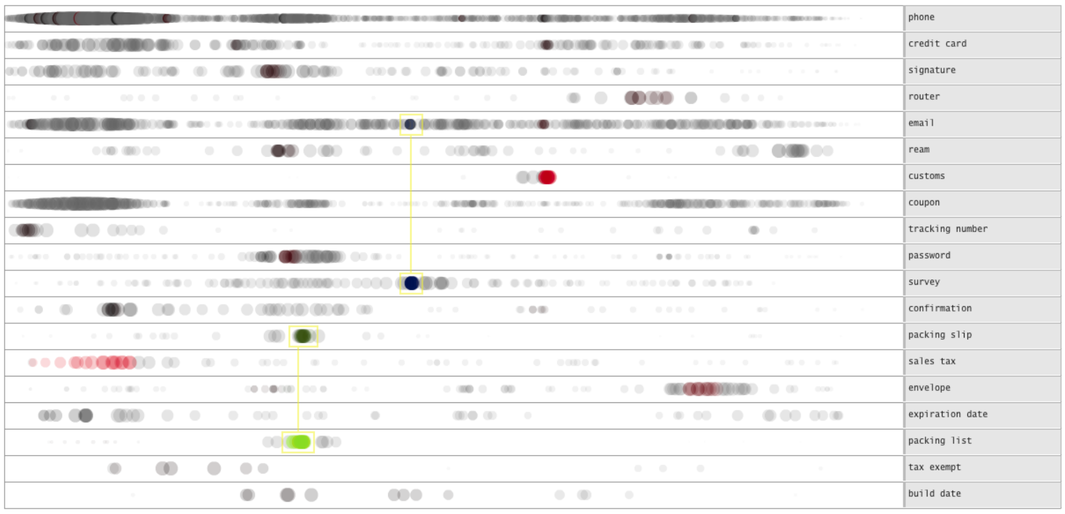
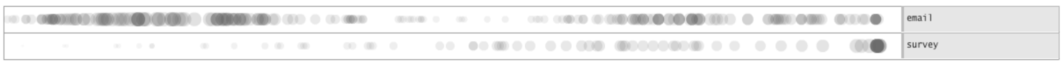


Figure 6.16: CloudLines showing a selection of event types from the customer feedback stream. Events belonging to important event episodes are red, while the similar event episodes are blue and green (highlighted with yellow outline).

quality of the output over time. The global correction using the offline kernel density estimation computation can be performed when the time and resource constraints allow it and it can include both numerical and visual evaluation of the stream processing output. In Figure 6.15, we show the differences that exist between the incremental and non-incremental version of the technique. This solution can be extended by employing a more sophisticated local adjustment criteria to ensure better quality control, which would, for example, take into account the distance from the center of the overlap area.

6.5.2 Exploration of Event Episodes Detected in Customer Surveys

Figure 6.16 shows the snapshot of the incremental CloudLines applied on the customer feedback stream, where the events that belong to the most significant event episodes are encoded with different colors. Additionally, the mapping in the HSV color model uses the value component to encode the event episode score, which is provided by the detection algorithm. The different colors show similar event episodes. Originally, all detected event episodes are red; when the user clicks on a specific episode, the episodes that are most similar and occur at the same time are painted with the same color (blue and green circles in the visualization). This image also shows two important aspects of the method: first, CloudLines helps in identifying longer temporal patterns within a single event sequence, while the online event episode detection algorithm identifies fine local features in these patterns that include not just the temporal dimension, but also the textual content of each event within an episode.



Topic 1: customer service department, service, phone, philippine, problem, hour, wrong, order, to order, frustrate, support staff
Topic 2: seem, work, to come, to spend, order, cover, waste, vista, tree, to seem, disappoint, system, laptop, many, not to get, to ask
Topic 3: survey, email, to send, same, product, to set, to take, monitor speaker, text, spam, email request, except, not to ship
Topic 4: federal express, package, to deliver, delivery, to think, block, home, fedex, to purchase, delivery requirement, shipping policy
Topic 5: owner, manual, product, not to receive, hard, to look, ohio, complain, to need, view, absence, mouse pad, restore disc

Figure 6.17: Real-time topic modeling for analytical reasoning. The two event episodes for email and survey appear in one topic.

The event detection algorithm first identifies interesting terms from the survey and each term represents one event type. Therefore, different event types may point to the same real-world issue and are also detected. Figure 6.17 shows the CloudLines for the terms *email* and *survey* and the output of the topic modeling algorithm that is immediately run on the similar documents. Several other topic keywords (to send, same, text, spam, email request) point to the issue: Due to an error, customers received the same *email* several times, which contained a request to participate in a *survey* about products and made them feel spammed.

Apart from the topics the analyst can also use the lens magnification feature of the CloudLine in order to get access to individual documents within dense areas and see their content using a tooltip, as shown in Figure 6.18.

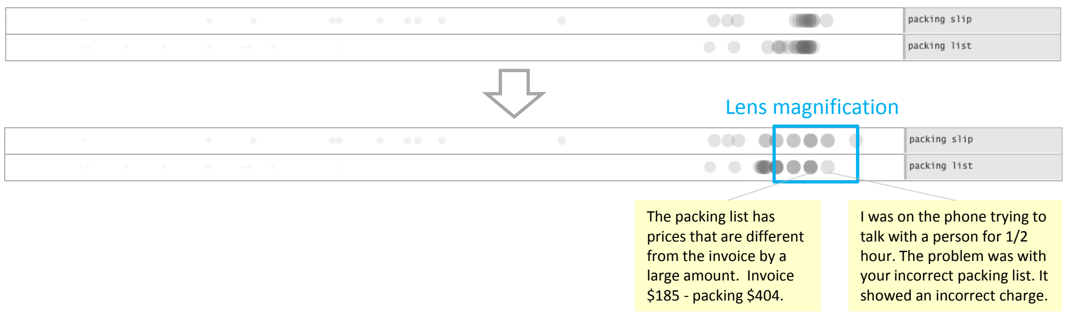

packing slip
packing list
Lens magnification
packing slip
packing list
The packing list has prices that are different from the invoice by a large amount. Invoice $185 - packing $404.
I was on the phone trying to talk with a person for 1/2 hour. The problem was with your incorrect packing list. It showed an incorrect charge.

Figure 6.18: The lens magnification makes documents visible without overlap in selected areas, so that their content can be displayed as a tooltip.

6.5.3 Applying Incremental CloudLines to Microblogs

The dataset is provided by the VAST Challenge 2011 Mini Challenge 1 and contains 1,023,056 messages collected April 30 - May 20, 2011 with timestamps and resolution of one minute. In order to validate the event detection algorithm and the visualization, the results are compared to the ground truth events from the VAST Challenge solution webpage[6]:

1. On May 17, a truck accident occurs on the I-610 Bridge on the VAST River (the cause of the disease).
2. Posts about inhalation cases on May 18th.
3. Posts about gastrointestinal cases caused by the spores from the water on May 19th.

The top issues detected by the algorithm are related to sickness symptoms. 25 representative event episodes are selected out of the 75 top episodes and visualized with CloudLines, with and without density distortion (see Figure 6.19). The 25 selected event episodes are marked in red. Sometimes the increase in frequency is so strong that only the first events are assigned to an episode and consequently only the outer left part of a dense area appears in red. In such cases, the expectation values quickly adapt to the change

[6]http://hcil.cs.umd.edu/localphp/hcil/vast11/index.php/solution/index

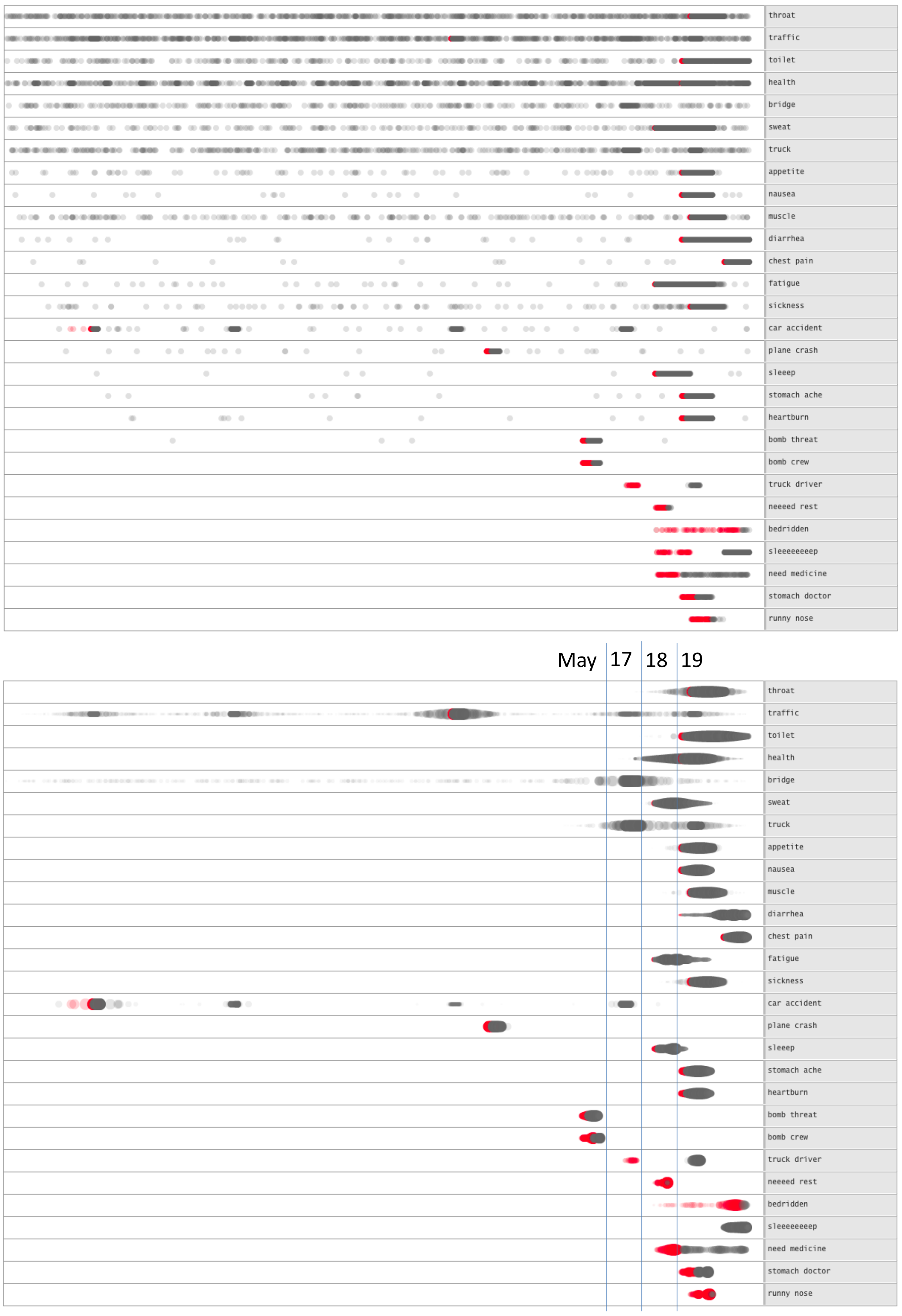


Figure 6.19: CloudLines without (top) and with (bottom) density distortion showing a selection of event types from the microblog stream. Events belonging to the top 75 automatically detected event episodes are marked in red.

in data load and after a short while consider the high data load to be expectable. The speed of adaption can be easily changed by tuning the parameter. Nevertheless, it shows that the algorithm is able to notify analysts about the initial phases of issues. The CloudLines visually depict the dynamics and the whole time range of the issue and the analyst can select it manually for further exploration.

In the visualization of Figure 6.19 three groups of event types can be distinguished visually and fit to the ground truth:

- Some event episodes refer to the truck accident on May 17th: *truck, bridge, truck driver, car accident, traffic*.
- Some event episodes refer to the flu-like symptoms on May 18th: *sweat, fatigue, need medicine, sleeep, neeeed rest, health*.
- Some event episodes refer to the gastrointestinal symptoms May 19th and 20th, for example: *stomach ache, appetite, toilet, stomach doctor, diarrhea*. Further symptoms like for example *runny nose*, problems with the *throat*, and later *chest pain*, also appear.
- There are some further event episodes not relating to the ground truth, but still interesting from an analysis perspective like *plane crash* or *bomb threat*.
- The event types *car accident* and *traffic* are nicely aligned, indicating a close relation that could be verified using the event episode similarity in the interactive exploration.

6.6 Summary

This chapter presented several experimental applications that integrate real time algorithms and streaming visualization in monitoring and detection tasks. The news stream retrieval system is described in detail, since its features and data properties are crucial for all approaches presented in this thesis. A detailed description and analysis of news and named entity metadata is given to illustrate the richness of the stream. I have presented a novel visual analytics algorithm called StreamingThreads for detection, tracking and visualization of event episodes in news streams with a forgetting model that includes aging of episodes. For the news monitoring tasks, the StreamSqueeze space-filling visualization technique is designed, which explores potential ways of dealing with conflicting criteria, such as layout stability preservation and visualization updates with dynamic streams in monitoring scenarios. Finally, an incremental version of the CloudLines visualization technique is presented at the end of the chapter and applied on two new incremental datasets. The techniques were applied in three domains: news data, customer feedback data and microblogging data. The different approaches show that the streaming visual analytics principles can be efficiently applied in different domains and used to solve different tasks.

7 Conclusion

Contents

In this thesis, I presented a novel visual analytics approach for analysis of events in incremental data sources and demonstrated how this approach can be applied in different usage contexts to support complex analytical tasks. I summarized design guidelines and challenges for *streaming visual analytics* to provide the right context for future researchers entering this complex but exciting field. The streaming visual analytics research framework led to the development of new methods for visual analysis of dynamic data and helped in creating more scalable analytic environments. I showed how we can improve existing visualizations, develop new ones, and integrate them with methods from information retrieval and human-computer interaction to support more effective analysis. I experimented with different visualization techniques to support detection, tracking and monitoring of events in real time and used them in different text stream domains to demonstrate their versatility. My work has contributed to the creation of a new, dynamic and exciting field at the crossroads of visualization, data mining and streaming data management.

7.1 Review of Contributions

This dissertation describes the following four contributions:

1. Streaming visual analytics research framework, which describes design considerations for developing visual analysis tools suitable for handling incremental data sources, and identifies challenges and issues in the user, data and visualization problem space;

2. CloudLines, a novel visualization method for analysis of temporal events in multiple sequences in limited space, which is coupled with interaction techniques for detailed exploration of events in data streams;

3. Story Tracker, a visual analytics system for analysis of text streams, which combines text clustering algorithms with incremental visualization to create a coherent analytical environment for analysis of news story development;

4. A set of extensions for real-time visual analytics for text streams, which demonstrate how streaming visualization techniques and event detection algorithms can be used in real-time scenarios.

7.1.1 Streaming Visual Analytics Research Framework

The review of relevant literature and my own experience led to the formulation of the design principles for streaming visual analytics. The primary focus was on identifying visualization challenges, defining design guidelines and considerations, and relating these to existing approaches in data stream management, data mining and human-computer interaction. I showed how *user interaction time* determines the selection of parameters for data processing algorithms and how the user feedback controls the *triggering of the updates*. Next, I described how *providing temporal context* and *updating strategies* efficiently inform the user about the changes in the data. In order to adapt existing visual analytics methods and data models to incremental environment, I proposed new *dynamic variables* - birth/death, age, rate of change and frequency. To deal with the information overflow in the visualization, I described the *object age relevance* criteria.

7.1.2 Interactive Exploration and Visualization of Event Episodes in Limited Space

I presented an interactive visualization method for detection and analysis of event episodes in multiple sequences of irregular event data. The method, called *CloudLines*, provides an overview of sequences while allowing direct interaction with each event in limited space. The method combines timeline distortion with density-based algorithms to help detecting important event episodes. A lens-based interaction allows direct access to overlapping events.

7.1.3 Visual Analysis for Incremental Exploration of Complex Story Evolution

Based on the streaming visual analytics principles, I designed an analytical framework for visual exploration of evolving text corpora. It is realized as a system for exploration of *dynamic* events in news streams whose type is not known in advance, *Story Tracker* and combines interactive visualization and text mining techniques to allow exploration of complex stories, i.e. stories that split and merge over time. I presented an incremental visualization to show evolution of stories and a novel interaction technique for filtering the stories based on *connectedness* and *duration* graph properties. The visualization can be modified with a sorting algorithm for multiple connected visual components, which minimizes the clutter and overlap from edge crossings. Story Tracker allows refinement of clustering parameters and incremental updates of both topics and visualization.

7.1.4 Real-time Techniques for Text Streams Monitoring

I experimented with different methods that combine real time data processing and incremental visualization applied on text streams and developed a visual analytics algorithm that detects, tracks and visualizes event episodes in news stream in real time with a forgetting model and incremental visualization updates. Next, I presented StreamSqueeze, a space-filling technique for monitoring news event data, which shows how to preserve layout stability when the stream is continuously updating. I also introduced the incremental extension of the CloudLines visualization, which is applied on customer survey text streams and microblog data to demonstrate its usefulness in other domains.

7.2 Limitations and Future Work

7.2.1 Modeling Streaming Visual Analytics Systems

At this point, the design considerations and identification of important challenges and issues in the data, visualization and user domains represent a first step towards a more formal definition of the model. My aim was to provide a systematic overview with practical advice based on the current approaches and the common problems I was facing while working with streaming data. The features of the data, visualization and user domain and their interactions create a complex problem space, where either a) there are many unknown aspects, or b) some aspects are impossible to measure. This complexity is also reflected in the fact that the visualization can display raw data, transformed data, as well as models that would have different data types and could change when the source data changes. A systematic classification of static visualization methods based on their visual features would help to identify appropriate streaming criteria. The next steps should fill these gaps.

7.2.2 The Role of the User and a Clear Definition of Tasks

Adaptive visual analytics systems that would react to changes in user behaviour and the stream are still very hard to formally define. A precise estimation of the user interaction time might be impossible in many cases. Another aspect is the separation of user roles. I designed my methods with expert users in mind, who understand the domain they want to analyze and also understand the mechanisms of the systems they are using. This integrated approach requires knowledge the analyst might not have. Therefore, a differentiation in user classes might help in solving this potential issue.

7.2.3 Variety of Real-time Visualization Methods

The nature of doing research in a new field requires trying out different approaches for different tasks, even if the tasks are in some ways similar. Also, my participation in several collaboration projects led to the development of different visualization techniques that could have relied on the same detection algorithms. The StreamingThreads algorithm, for example, could have been tested with the incremental extension of the CloudLines technique. On the other hand, these collaboration projects allowed me to test my concepts on new datasets and in different domains, which demonstrated the applicability of my methods.

7.2.4 Evaluation

For the most of the presented methods, I have used case studies to show their usefulness. The complexity of the visual analytics systems presented in this thesis, which includes incremental data, semi-structured text and different visualization techniques makes it very difficult to evaluate them differently. To the best of my knowledge, there are no benchmarks for this type of problems and it is questionable how these benchmarks should be created. I have attempted to elaborate in detail every design decision I made during the development process, which consisted of many iterations, improvements and alternative solutions.

7.3 Closing Remarks

Streaming visual analytics is a new and challenging field, which tackles complex problems in data, visualization and user spaces and opens huge opportunities for future research. Future work should address the challenges that occur in transition stages between the updates, the propagation of streaming data through the information visualization pipeline and process times needed to get from the raw data to the visualization, the interplay between user interaction and automatic updates, but also the different types of updates (automatic and user triggered). Finally, an important research direction is how to deal with the visualization when the volume of the data is too big and data approximation occurs.

Although this research is in its inception phase, I believe that my contributions will serve as a sound basis for future research in this challenging domain. In the end, we need a strong streaming visual analytics model that would integrate data properties and features of the visual objects in the context of transient data streams to enable a faster and tighter human-computer integration in the future systems.

List of Figures

List of Tables

Bibliography

[1] C. Aggarwal. *Data streams: models and algorithms*. Springer, New York, 2007.

[2] C. Aggarwal, J. Han, J. Wang, and P. Yu. On demand classification of data streams. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 503–508. ACM, 2004.

[3] W. Aigner, S. Miksch, W. Muller, H. Schumann, and C. Tominski. Visualizing time-oriented data–A systematic view. *Computers & Graphics*, 31(3):401–409, 2007.

[4] W. Aigner, S. Miksch, W. Muller, H. Schumann, and C. Tominski. Visual methods for analyzing time-oriented data. *Visualization and Computer Graphics, IEEE Transactions on*, 14(1):47–60, 2008.

[5] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-oriented Data*. Human-Computer Interaction Series. Springer-Verlag New York Inc, 2011.

[6] C. Albrecht-Buehler, B. Watson, and D. Shamma. Visualizing live text streams using motion and temporal pooling. *IEEE Computer Graphics and Applications*, 25(3):52–59, 2005.

[7] C. Albrecht-Buehler, B. Watson, and D. A. Shamma. TextPool: Visualizing Live Text Streams. In *Proceedings of the IEEE Symposium on Information Visualization*, page 215.1, Washington, DC, USA, 2004. IEEE Computer Society.

[8] J. Allan, J. Carbonell, G. Doddington, J. Yamron, Y. Yang, J. A. Umass, B. A. Cmu, D. B. Cmu, A. B. Cmu, R. B. Cmu, I. C. Dragon, G. D. Darpa, A. H. Cmu, J. L. Cmu, V. L. Umass, X. L. Cmu, S. L. Dragon, P. V. M. Dragon, R. P. Umass, T. P. Cmu, J. P. Umass, and M. S. Umass. Topic Detection and Tracking Pilot Study Final Report. In *In Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, 1998.

[9] J. Alsakran, Y. Chen, D. Luo, Y. Zhao, J. Yang, W. Dou, and S. Liu. Real-time visualization of streaming text with force-based dynamic system. *Computer Graphics and Applications, IEEE*, 32(1):34–45, 2012.

[10] J. Alsakran, Y. Chen, Y. Zhao, J. Yang, and D. Luo. Streamit: Dynamic visualization and interactive exploration of text streams. In *Pacific Visualization Symposium (PacificVis), 2011 IEEE*, pages 131–138. IEEE, 2011.

[11] L. AlSumait, D. Barbará, J. Gentle, and C. Domeniconi. Topic significance ranking of lda generative models. In *Machine Learning and Knowledge Discovery in Databases*, pages 67–82. Springer, 2009.

[12] M. Ankerst, D. A. Keim, and H.-P. Kriegel. Circle segments: A technique for visually exploring large multidimensional data sets. In *Visualization '96, Hot Topic Session, San Francisco, CA*, 1996.

[13] M. Atkinson and E. Van der Goot. Near real time information mining in mulitlingual news. In *WWW '09: Proceedings of the 18th international conference on World Wide Web*, pages 1153–1154. ACM, 2009.

[14] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '02, pages 1–16, New York, NY, USA, 2002. ACM.

[15] P. Bak, F. Mansmann, H. Janetzko, and D. A. Keim. Spatiotemporal analysis of sensor logs using growth ring maps. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 2009.

[16] F. Beck, M. Burch, and S. Diehl. Towards an aesthetic dimensions framework for dynamic graph visualisations. In *Information Visualisation, 2009 13th International Conference*, pages 592–597. IEEE, 2009.

[17] B. B. Bederson, J. Grosjean, and J. Meyer. Toolkit design for interactive structured graphics. *IEEE Trans. Softw. Eng.*, 30:535–546, August 2004.

[18] B. B. Bederson, B. Shneiderman, and M. Wattenberg. Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. *AcM Transactions on Graphics (TOG)*, 21(4):833–854, 2002.

[19] F. Bendix, R. Kosara, and H. Hauser. Parallel sets: visual analysis of categorical data. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 133 – 140, 2005.

[20] L. Berry and T. Munzner. Binx: Dynamic exploration of time series datasets across aggregation levels. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages p2–p2. IEEE, 2004.

[21] E. Bertini, P. Hertzog, and D. Lalanne. SpiralView: towards security policies assessment through visual correlation of network resources with evolution of alarms. In *IEEE Symposium on Visual Analytics Science and Technology, 2007. VAST 2007*, pages 139–146, 2007.

[22] D. Best, R. Hafen, B. Olsen, and W. Pike. Atypical behavior identification in large-scale network traffic. In *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, pages 15–22. IEEE, 2011.

[23] D. M. Best, S. Bohn, D. Love, A. Wynne, and W. A. Pike. Real-time visualization of network behaviors for situational awareness. In *Proceedings of the Seventh International Symposium on Visualization for Cyber Security*, VizSec '10, pages 79–90, New York, NY, USA, 2010. ACM.

[24] C. Binucci, U. Brandes, G. Di Battista, W. Didimo, M. Gaertler, P. Palladino, M. Patrignani, A. Symvonis, and K. Zweig. Drawing trees in a streaming model. In *17th International Symposium on Graph Drawing*, pages 292–303. Springer, 2010.

[25] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *NIPS*, volume 16, 2003.

[26] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 113–120, New York, NY, USA, 2006. ACM.

[27] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[28] M. Bruls, K. Huizing, and J. J. Van Wijk. Squarified treemaps. In *Data Visualization 2000*, pages 33–42. Springer, 2000.

[29] R. Budiu, C. Royer, and P. Pirolli. Modeling information scent: A comparison of lsa, pmi and glsa similarity measures on common tests and corpora. In *Large Scale Semantic Access to Content (Text, Image, Video, and Sound)*,

pages 314–332. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, 2007.

[30] L. Byron and M. Wattenberg. Stacked graphs–geometry & aesthetics. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1245–1252, 2008.

[31] S. Card, J. Mackinlay, and B. Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[32] M. S. T. Carpendale and C. Montagnese. A framework for unifying presentation space. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, UIST '01, pages 61–70, New York, NY, USA, 2001. ACM.

[33] S. Carpendale, J. Ligh, and E. Pattison. Achieving higher magnification in context. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 71–80, New York, NY, USA, 2004. ACM.

[34] A. J.-B. Chaney and D. M. Blei. Visualizing topic models. In *ICWSM*, 2012.

[35] J. Chang, J. L. Boyd-Graber, S. Gerrish, C. Wang, and D. M. Blei. Reading tea leaves: How humans interpret topic models. In *NIPS*, volume 22, pages 288–296, 2009.

[36] N. Chaudhry, K. Shaw, and M. Abdelguerfi. *Stream data management*, volume 30. Springer Verlag, 2005.

[37] C. Chen. Top 10 unsolved information visualization problems. *Computer Graphics and Applications, IEEE*, 25(4):12–16, 2005.

[38] G. Chin, M. Singhal, G. Nakamura, V. Gurumoorthi, and N. Freeman-Cadoret. Visual analysis of dynamic data streams. *Information Visualization*, 8(3):212–229, 2009.

[39] J. Chuang, S. Gupta, C. Manning, and J. Heer. Topic model diagnostics: Assessing domain relevance via topical alignment. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 612–620, 2013.

[40] J. Chuang, C. D. Manning, and J. Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 74–77. ACM, 2012.

[41] J. Chuang, D. Ramage, C. Manning, and J. Heer. Interpretation and trust: Designing model-driven visualizations for text analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 443–452. ACM, 2012.

[42] C. Collins, F. B. Viégas, and M. Wattenberg. Parallel Tag Clouds to explore and analyze faceted text corpora. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST 2009)*, pages 91–98, 2009.

[43] C. Collins, F. B. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, pages 91–98. IEEE, 2009.

[44] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong. Textflow: Towards better understanding of evolving topics in text. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2412–2421, 2011.

[45] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu. Context preserving dynamic word cloud visualization. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE*, pages 121–128. IEEE, 2010.

[46] O. Daae Lampe and H. Hauser. Interactive visualization of streaming data with kernel density estimation. In *Pacific Visualization Symposium (PacificVis)*, pages 171–178. IEEE, 2011.

[47] N. Diakopoulos, M. Naaman, and F. Kivran-Swaine. Diamonds in the rough: Social media visual analytics for journalistic inquiry. In *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, pages 115–122. IEEE, 2010.

[48] D. DiBiase, A. MacEachren, J. Krygier, and C. Reeves. Animation and the role of map design in scientific visualization. *Cartography and geographic information science*, 19(4):201–214, 1992.

[49] M. Dork, D. Gruen, C. Williamson, and S. Carpendale. A visual backchannel for large-scale events. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1129–1138, 2010.

[50] W. Dou, X. Wang, R. Chang, and W. Ribarsky. Paralleltopics: A probabilistic approach to exploring document collections. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 231–240. IEEE, 2011.

[51] W. Dou, X. Wang, D. Skau, W. Ribarsky, and M. X. Zhou. Leadline: Interactive visual analysis of text data through event identification and exploration. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 93–102. IEEE, 2012.

[52] W. Dou, L. Yu, X. Wang, Z. Ma, and W. Ribarsky. Hierarchicaltopics: Visually exploring large text collections using topic hierarchies. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2002–2011, 2013.

[53] M. Dubinko, R. Kumar, J. Magnani, J. Novak, P. Raghavan, and A. Tomkins. Visualizing tags over time. *ACM Transactions of the Web (TWEB)*, 1, August 2007.

[54] S. Eick and A. Karr. Visual scalability. *Journal of Computational and Graphical Statistics*, 11(1):22–43, 2002.

[55] G. Ellis and A. Dix. A taxonomy of clutter reduction for information visualisation. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1216–1223, 2007.

[56] M. Endsley. Toward a Theory of Situation Awareness in Dynamic Systems. *Human Factors: The Journal of the Human*, 37(1):32–64, 1995.

[57] D. Fisher, A. Hoff, G. Robertson, and M. Hurst. Narratives: A visualization to track narrative events as they develop. In *IEEE Symposium on Visual Analytics Science and Technology, 2008. VAST '08*, pages 115–122, 2008.

[58] D. Fisher, I. Popov, S. Drucker, and m. schraefel. Trust me, i'm partially right: incremental visualization lets analysts explore large datasets faster. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1673–1682, New York, NY, USA, 2012. ACM.

[59] A. Forbes, T. Hollerer, and G. Legrady. behaviorism: a framework for dynamic data visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1164–1171, 2010.

[60] S. Foresti, J. Agutter, Y. Livnat, S. Moon, and R. Erbacher. Visual correlation of network alerts. *IEEE Computer Graphics and Applications*, 26:48–59, 2006.

[61] C. Franke, M. Karnstedt, and K. Sattler. Mining data streams under dynamicly changing resource constraints. *KDML: Knowledge Discovery, Data Mining, and Machine Learning*, pages 262–269, 2006.

[62] G. W. Furnas. A fisheye follow-up: further reflections on focus + context. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 999–1008, New York, NY, USA, 2006. ACM.

[63] M. Gaber, A. Zaslavsky, and S. Krishnaswamy. Mining data streams: a review. *ACM Sigmod Record*, 34(2):18–26, 2005.

[64] J. Gama. *Knowledge discovery from data streams*. Data Mining and Knowledge Discovery Series. Chapman & Hall, CRC Press, 2010.

[65] E. R. Gansner, Y. Hu, and S. North. Visualizing streaming text data with dynamic graphs and maps. In *Graph Drawing*, pages 439–450. Springer, 2013.

[66] M. Ghoniem, D. Luo, J. Yang, and W. Ribarsky. Newslab: Exploratory broadcast news video analysis. In *VAST '07: Proceedings of the 2007 IEEE Symposium on Visual Analytics Science and Technology*, pages 123–130. IEEE Computer Society, 2007.

[67] L. Golab and M. Özsu. *Data Stream Management*. Morgan & Claypool Publishers, 2010.

[68] L. Golab and M. T. Özsu. Issues in data stream management. *SIGMOD Rec.*, 32(2):5–14, June 2003.

[69] B. Gretarsson, J. O'donovan, S. Bostandjiev, T. Höllerer, A. Asuncion, D. Newman, and P. Smyth. Topicnets: Visual analysis of large text corpora with topic modeling. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(2):23, 2012.

[70] M. Grčar, V. Podpečan, M. Juršič, and N. Lavrač. Efficient visualization of document streams. In *Proceedings of the 13th international conference on Discovery science*, DS'10, pages 174–188, Berlin, Heidelberg, 2010. Springer-Verlag.

[71] S. Guha, N. Koudas, and K. Shim. Data-streams and histograms. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 471–475. ACM, 2001.

[72] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2011.

[73] M. Hao, U. Dayal, D. Keim, and T. Schreck. Multi-resolution techniques for visual exploration of large time-series data. In *Eurographics/IEEE-VGTC Symposium on Visualization*, pages 27–34. The Eurographics Association, 2007.

[74] M. C. Hao, D. A. Keim, U. Dayal, D. Oelke, and C. Tremblay. Density Displays for Data Stream Monitoring. *Computer Graphics Forum*, 27(3):895–902, 2008.

[75] M. C. Hao, D. A. Keim, U. Dayal, and T. Schreck. Importance-driven visualization layouts for large time series data. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis '05)*, 2005.

[76] M. Harrower and C. A. Brewer. *ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps*, pages 261–268. John Wiley and Sons, Ltd, 2011.

[77] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. Themeriver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002.

[78] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: the effects of chart size and layering on the graphical perception of time series visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1303–1312. ACM, 2009.

[79] J. Heer and G. Robertson. Animated transitions in statistical data graphics. *IEEE transactions on visualization and computer graphics*, 13(6):1240–1247, 2007.

[80] M. Hegarty. Dynamic visualizations and learning: Getting to the difficult questions. *Learning and Instruction*, 14(3):343–352, 2004.

[81] E. G. Hetzler, V. L. Crow, D. A. Payne, and A. E. Turner. Turning the bucket of text into a pipe. In *INFOVIS '05: Proceedings of the 2005 IEEE Symposium on Information Visualization*, page 12. IEEE Computer Society, 2005.

[82] H. Hochheiser and B. Shneiderman. Dynamic query tools for time series data sets: timebox widgets for interactive exploration. *Information Visualization*, 3(1):1, 2004.

[83] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *IEEE Visualization*, pages 361–378, 1990.

[84] Y. Ishikawa and M. Hasegawa. T-Scroll: Visualizing Trends in a Time-Series of Documents for Interactive User Exploration. In L. Kovács, N. Fuhr, and C. Meghini, editors, *Research and Advanced Technology for Digital Libraries*, volume 4675 of *Lecture Notes in Computer Science*, pages 235–246. Springer Berlin / Heidelberg, 2007.

[85] P. Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[86] W. Javed, B. McDonnel, and N. Elmqvist. Graphical perception of multiple time series. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):927–934, 2010.

[87] W. Javed, B. McDonnel, and N. Elmqvist. Graphical Perception of Multiple Time Series. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):927–934, 2010.

[88] B. Johnson and B. Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Visualization, 1991. Visualization'91, Proceedings., IEEE Conference on*, pages 284–291. IEEE, 1991.

[89] O. Kaser and D. Lemire. Tag-cloud drawing: Algorithms for cloud visualization. *arXiv preprint cs/0703109*, 2007.

[90] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. *Visual analytics: Definition, process, and challenges*. Springer, 2008.

[91] D. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Challenges in visual data analysis. In *Information Visualization, 2006. IV 2006. Tenth International Conference on*, pages 9–16. IEEE, 2006.

[92] D. A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *Visualization and Computer Graphics, IEEE Transactions on*, 6(1):59–78, 2000.

[93] D. A. Keim. Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8, 2002.

[94] D. A. Keim, M. Ankerst, and H.-P. Kriegel. Recursive pattern: A technique for visualizing very large amounts of data. In *Proceedings of the 6th conference on Visualization'95*, page 279. IEEE Computer Society, 1995.

[95] D. A. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, editors. *Mastering The Information Age - Solving Problems with Visual Analytics*. Eurographics, 2010.

[96] D. A. Keim, M. Krstajic, C. Rohrdantz, and T. Schreck. Real-time Visual Analytics of Text Data Streams. *IEEE Computer*, 46(7):47–55, 2013.

[97] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. Visual Analytics: Scope and Challenges. In S. Simoff, M. H. Boehlen, and A. Mazeika, editors, *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*. Springer, 2008. Lecture Notes in Computer Science (LNCS).

[98] D. A. Keim, J. Schneidewind, and M. Sips. Circleview - a new approach for visualizing time related multidimensional data sets. In *ACM Advanced Visual Interfaces, International Working Conference, AVI 2004, May 25-28, Gallipoli (Lecce), Italy*. Association for Computing Machinery (ACM), ACM Press, May 2004.

[99] B. Kerr. THREAD ARCS: An Email Thread Visualization. *Information Visualization, IEEE Symposium on*, 0:27–218, 2003.

[100] R. Kincaid. Line graph explorer: scalable display of line graphs using focus+context. In *In Working Conference on Advanced Visual interfaces*, pages 404–411. ACM Press, 2006.

[101] R. Kincaid. Signallens: Focus+context applied to electronic time series. *IEEE Transactions on Visualization and Computer Graphics*, 16:900–907, November 2010.

[102] J. Kleinberg. Bursty and hierarchical structure in streams. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 91–101. ACM, 2002.

[103] J. Kleinberg. *Temporal Dynamics of On-Line Information Streams*. Springer, 2006.

[104] M. Krstajic, E. Bertini, and D. A. Keim. CloudLines: Compact Display of Event Episodes in Multiple Time-Series. *IEEE Transactions on Visualization and Computer Graphics*, 17:2432–2439, 2011.

[105] M. Krstajic, E. Bertini, F. Mansmann, and D. A. Keim. Visual analysis of news streams with article threads. In *StreamKDD '10: Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques*, pages 39–46, New York, NY, USA, 2010. ACM.

[106] M. Krstajic and D. A. Keim. Visualization of streaming data: Observing change and context in information visualization techniques. In *Big Data, 2013 IEEE International Conference on*, pages 41–47. IEEE, 2013.

[107] M. Krstajic, F. Mansmann, A. Stoffel, M. Atkinson, and D. Keim. Processing online news streams for large-scale semantic analysis. In *1st International Workshop on Data Engineering meets the Semantic Web*, 2010.

[108] M. Krstajic, M. Najm-Araghi, F. Mansmann, and D. A. Keim. Incremental Visual Text Analytics of News Story Development. In *Proceedings of Conference on Visualization and Data Analysis (VDA '12), Best Paper Award*, 2012.

[109] M. Krstajić, M. Najm-Araghi, F. Mansmann, and D. A. Keim. Story tracker: Incremental visual text analytics of news story development. *Information Visualization*, 12(3-4):308–323, 2013.

[110] M. Krstajic, C. Rohrdantz, M. Hund, and A. Weiler. *Getting there first: Real-time detection of real-world incidents on twitter*. Bibliothek der Universität Konstanz, 2012.

[111] N. Kumar, N. Lolla, E. Keogh, S. Lonardi, and C. A. Ratanamahatana. Time-series bitmaps: a practical visualization tool for working with large time series databases. In *SIAM 2005 Data Mining Conference*, pages 531–535. SIAM, 2005.

[112] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, and J. Allan. Mining of Concurrent Text and Time Series. In *In proceedings of the 6 th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining Workshop on Text Mining*, pages 37–44, 2000.

[113] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[114] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, page 11. ACM, 2003.

[115] X. Liu, Y. Hu, S. North, and H.-W. Shen. Compactmap: A mental map preserving visual interface for streaming text data. In *Big Data, 2013 IEEE International Conference on*, pages 48–55. IEEE, 2013.

[116] L. Lloyd, D. Kechagias, and S. Skiena. Lydia: A system for large-scale news analysis. In *String Processing and Information Retrieval: 12th International Conference, SPIRE 2005, Buenos Aires, Argentina, November 2-4, 2005: Proceedings*, pages 161–166, 2005.

[117] D. Luo, J. Yang, M. Krstajic, W. Ribarsky, and D. Keim. Eventriver: Visually exploring text collections with temporal references. *IEEE Transactions on Visualization and Computer Graphics*, 99(PrePrints), 2010.

[118] A. MacEachren. *How maps work: representation, visualization, and design*. The Guilford Press, 2004.

[119] F. Mansmann, F. Fischer, and D. A. Keim. Dynamic Visual Analytics - Facing the Real-Time Challenge. In J. Dill, R. A. Earnshaw, D. J. Kasik, J. A. Vince, and P. C. Wong, editors, *Expanding the Frontiers of Visual Analytics and Visualization, to appear*. Springer, 2012.

[120] F. Mansmann, M. Krstajic, F. Fischer, and E. Bertini. StreamSqueeze: A Dynamic Stream Visualization for Monitoring of Event Data. In *Proceedings of Conference on Visualization and Data Analysis (VDA '12), to appear*, 2012.

[121] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 227–236. ACM, 2011.

[122] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. LiveRAC: interactive visual exploration of system management time-series data. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1483–1492. ACM, 2008.

[123] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 262–272. Association for Computational Linguistics, 2011.

[124] R. Motwani, J. Widom, A. Arasu, B. Babcock, S. Babu, M. Datar, G. Manku, C. Olston, J. Rosenstein, and R. Varma. Query processing, resource management, and approximation ina data stream management system. In *CIDR 2003*. Stanford InfoLab, 2002.

[125] W. Müller and H. Schumann. Visualization for modeling and simulation: visualization methods for time-dependent data - an overview. In *Proceedings of the 35th conference on Winter simulation: driving innovation*, WSC '03, pages 737–745. Winter Simulation Conference, 2003.

[126] C. C. Musat, J. Velcin, S. Trausan-Matu, and M.-A. Rizoiu. Improving topic evaluation using conceptual knowledge. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 1866–1871. AAAI Press, 2011.

[127] D. Newman, C. Chemudugunta, P. Smyth, and M. Steyvers. Analyzing entities and topics in news articles using statistical topic models. In *Intelligence and Security Informatics*, pages 93–104. Springer, 2006.

[128] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. Association for Computational Linguistics, 2010.

[129] S. Osinski, J. Stefanowski, and D. Weiss. Lingo: Search results clustering algorithm based on singular value decomposition. In *Intelligent Information Systems*, pages 359–368, 2004.

[130] S. Osinski and D. Weiss. Carrot^2: Design of a flexible and efficient web information retrieval framework. In *AWIC*, pages 439–444, 2005.

[131] T. Palpanas, M. Vlachos, E. Keogh, and D. Gunopulos. Streaming time series summarization using user-defined amnesic functions. *IEEE Trans. on Knowl. and Data Eng.*, 20:992–1006, July 2008.

[132] E. Pietriga and C. Appert. Sigma lenses: focus-context transitions combining space, time and translucence. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1343–1352, New York, NY, USA, 2008. ACM.

[133] B. Pouliquen, M. Kimler, R. Steinberger, C. Ignat, T. Oellinger, K. Blackler, F. Fuart, W. Zaghouani, A. Widiger, A.-C. Forslund, and C. Best. Geocoding multilingual texts: Recognition, disambiguation and visualisation. In *Proceedings of LREC-2006*, Sep 2006.

[134] D. Ramage, S. T. Dumais, and D. J. Liebling. Characterizing microblogs with topic models. In *ICWSM*, 2010.

[135] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 248–256. Association for Computational Linguistics, 2009.

[136] R. Rea. Ibm infosphere streams: Redefining real time analytic processing. *IBM White Paper*, 2010.

[137] C. Reas and B. Fry. *Processing: a programming handbook for visual designers and artists*. The MIT Press, 2007.

[138] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 995–998. ACM, 2007.

[139] G. Robertson, D. Ebert, S. Eick, D. Keim, and K. Joy. Scale and complexity in visual analytics. *Information Visualization*, 8(4):247–253, 2009.

[140] C. Rohrdantz. *Visual Analytics of Change in Natural Language*. PhD thesis, University of Konstanz, 2014. Ph.D. Dissertation.

[141] C. Rohrdantz, M. C. Hao, U. Dayal, L.-E. Haug, and D. A. Keim. Feature-based visual sentiment analysis of text document streams. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(2):26, 2012.

[142] C. Rohrdantz, D. Oelke, M. Krstajic, and F. Fischer. Real-Time Visualization of Streaming Text Data: Tasks and Challenges. In *Workshop on Interactive Visual Text Analytics for Decision-Making at the IEEE VisWeek 2011*, 2011.

[143] G. E. Rosario, E. A. Rundensteiner, D. C. Brown, and M. O. Ward. Mapping nominal values to numbers for effective visualization. In *Information Visualization, 2003. INFOVIS 2003. IEEE Symposium on*. IEEE Computer Society, 2003.

[144] S. J. Rose, S. Butner, W. Cowley, M. L. Gregory, and J. Walker. Describing story evolution from dynamic information streams. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST '09)*, pages 99–106, 2009.

[145] S. J. Rose, S. Butner, W. Cowley, M. L. Gregory, and J. Walker. Describing story evolution from dynamic information streams. In *IEEE Symposium on Visual Analytics Science and Technology, 2008. VAST '09*, pages 99–106. IEEE, 2009.

[146] R. ROSENBAUM and H. SCHUMANN. Progressive refinementmore than a means to overcome limited bandwidth. In *Proceedings of Conference on Visualization and Data Analysis (VDA)*. Society of Photo-Optical Instrumentation Engineers, 2009.

[147] T. Saito, H. N. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya, and T. Kaseda. Two-tone pseudo coloring: Compact visualization for one-dimensional data. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 173–180. IEEE, 2005.

[148] T. Saito, H. N. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya, and T. Kaseda. Two-tone pseudo coloring: Compact visualization for one-dimensional data.

In *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, pages 23–. IEEE Computer Society, 2005.

[149] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

[150] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[151] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18:613–620, November 1975.

[152] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer. On the beauty and usability of tag clouds. In *Information Visualisation, 2008. IV'08. 12th International Conference*, pages 17–25. IEEE, 2008.

[153] B. Shneiderman. Dynamic queries for visual information seeking. *Software, IEEE*, 11(6):70–77, 1994.

[154] B. Shneiderman and M. Wattenberg. Ordered treemap layouts. In *Information Visualization, 2001. INFOVIS 2001. IEEE Symposium on*, pages 73–78. Ieee, 2001.

[155] B. W. Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

[156] D. Simons and R. Rensink. Change blindness: Past, present, and future. *Trends in cognitive sciences*, 9(1):16–20, 2005.

[157] R. Spence. *Information visualization: Design for interaction*. Prentice Hall, 2007.

[158] J. Stefanowski and D. Weiss. Carrot and language properties in web search results clustering. In *AWIC*, pages 240–249, 2003.

[159] R. Steinberger and B. Pouliquen. Cross-lingual named entity recognition. *Linguisticae Investigationes*, 30(1):135–162, January 2007.

[160] K. Stevens, P. Kegelmeyer, D. Andrzejewski, and D. Buttler. Exploring topic coherence over many models and many topics. In *Proceedings of the 2012 Joint*

Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 952–961. Association for Computational Linguistics, 2012.

[161] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315. ACM, 2004.

[162] K. Sugiyama, S. Tagawa, and M. Toda. Methods for Visual Understanding of Hierarchical System Structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, 1981.

[163] T. Takada and H. Koike. Mielog: A highly interactive visual log browser using information visualization and statistical analysis. In *In Proc. USENIX Conf. on System Administration*, pages 133–144, 2002.

[164] H. Tanev, J. Piskorski, and M. Atkinson. Real-Time News Event Extraction for Global Crisis Monitoring. In *Proceedings of the 13th International Conference on Applications of Natural Language to Information Systems (NLDB 2008), London, UK*, pages 24–27. Springer, 2008.

[165] S. T. Teoh and K.-L. Ma. Paintingclass: interactive construction, visualization and exploration of decision trees. In *KDD*, pages 667–672. ACM, 2003.

[166] J. Thomas and K. Cook. *Illuminating the path: The research and development agenda for visual analytics*. IEEE Computer Society, 2005.

[167] I. Titov and R. McDonald. A joint model of text and aspect ratings for sentiment summarization. *Urbana*, 51:61801, 2008.

[168] Y. Tu and H.-W. Shen. Visualizing changes of hierarchical data using treemaps. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1286–1293, 2007.

[169] D. Turo and B. Johnson. Improving the visualization of hierarchies with treemaps: design issues and experimentation. In *Visualization, 1992. Visualization'92, Proceedings., IEEE Conference on*, pages 124–131. IEEE, 1992.

[170] B. Tversky, J. Morrison, and M. Betrancourt. Animation: can it facilitate? *International journal of human-computer studies*, 57(4):247–262, 2002.

[171] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. W. Fellner. Visual analysis of large graphs: State-of-the-art and future research challenges. In *Computer graphics forum*, volume 30, pages 1719–1749. Wiley Online Library, 2011.

[172] H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1105–1112. ACM, 2009.

[173] M. P. Wand and M. C. Jones. *Kernel Smoothing (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. Chapman and Hall/CRC, 1 edition, Dec. 1994.

[174] X. Wang and A. McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433. ACM, 2006.

[175] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[176] M. Wattenberg. Baby names, visualization, and social data analysis. In *INFOVIS '05: Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, page 1, Washington, DC, USA, 2005. IEEE Computer Society.

[177] F. Wei, S. Liu, Y. Song, S. Pan, M. X. Zhou, W. Qian, L. Shi, L. Tan, and Q. Zhang. Tiara: a visual exploratory text analytic system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 153–162, New York, NY, USA, 2010. ACM.

[178] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185. ACM, 2006.

[179] M. Weskamp. Newsmap. *Webdesigning Magazine*, June 2004. http://www.newsmap.jp.

[180] Wikipedia. Kernel density estimate, 2011. http://www.wikipedia.org, accessed on July 1, 2011.

[181] J. Wise, J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. *Information Visualization, IEEE Symposium on*, page 51, 1995.

[182] J. A. Wise. The ecological approach to text visualization. *JASIS*, 50(13):1224–1233, 1999.

[183] P. C. Wong, H. Foote, D. Adams, W. Cowley, and J. Thomas. Dynamic visualization of transient data streams. In *IEEE Symposium on Information Visualization (INFOVIS 2003)*, volume 0, page 13, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[184] Z. Xie. Towards exploratory visualization of multivariate streaming data. In *IEEE Vis/InfoVis/VAST Doctoral Colloquium*, 2007.

[185] Z. Xie. *Exploratory Visualization of Data Pattern Changes in Multivariate Data Streams*. PhD thesis, Worcester Polytechnic Institute, 2011.

[186] Z. Xie, M. Ward, and E. Rundensteiner. Visual analysis of multivariate data streams based on doi functions. Technical report, Technical Report TR-10-06, Worcester Polytechnic Institute, Computer Science Department, 2010.

[187] Z. Xie, M. Ward, and E. Rundensteiner. Visual exploration of stream pattern changes using a data-driven framework. In *Advances in Visual Computing, Second International Symposium, ISVC 2006, Lake Tahoe, NV, USA*, pages 522–532. Springer, 2010.

[188] D. Yang, Z. Guo, Z. Xie, E. Rundensteiner, and M. Ward. Interactive visual exploration of neighbor-based patterns in data streams. In *Proceedings of the 2010 international conference on Management of data*, pages 1151–1154. ACM, 2010.

[189] D. Yang, E. Rundensteiner, and M. Ward. Neighbor-based pattern detection for windows over streaming data. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology (EDBT)*, pages 529–540. ACM, 2009.

[190] D. Yang, E. Rundensteiner, and M. Ward. Summarization and matching of complex patterns in streaming environment. Technical report, WPI-CS-TR-11, 2011.

[191] D. Yang, E. Rundensteiner, and M. Ward. Shared execution strategy for neighbor-based pattern mining requests over streaming windows. *ACM Transactions on Database Systems (TODS)*, 37(1):5, 2012.

[192] D. Yang, E. A. Rundensteiner, and M. O. Ward. Summarization and matching of density-based clusters in streaming environments. *PVLDB vol. 5 / VLDB*, pages 121–132, October 2011.

[193] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *ACM International Conference on Web Search and Data Minig (WSDM)*. Stanford InfoLab.

[194] E. A. R. Zaixian Xie, Matthew O. Ward. Exploring multivariate data streams using windowing and sampling strategies. *CHI: Interacting with temporal data workshop*, 2009.

[195] O. E. Zamir. Clustering web documents: A phrase-based method for grouping search engine results. Technical report, 1999.

[196] J. Zhang, Y. Song, C. Zhang, and S. Liu. Evolutionary hierarchical dirichlet processes for multiple correlated time-varying corpora. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1079–1088, New York, NY, USA, 2010. ACM.

[197] I. Zliobaite and M. Pechenizkiy. Reference framework for handling concept drift: An application perspective. 2010.