ABSTRACT
As of October 2011, Twitter users were producing approximately 250 million tweets per day [12]. With this increase in usage, Twitter has become a rich resource for social information and crowdsourced data. Researchers in human-computer interaction (HCI) have become increasingly interested in the analysis and visualization of social media [6]. Current visualization tools for exploring Twitter data are mostly limited to hashtag and word frequencies, and very few take advantage of the tweets as a natural language corpus.

In this paper, we present two main contributions. First, we extended the tweet-collection functionality of NodeXL, a network visualization tool using Microsoft Excel, to allow aggregation of summary statistics (such as word and word pair frequencies) across multiple workbooks into a single workbook. Second, we developed a tool that visualizes the results of automatic topic detection and topic alignment between sets of tweets over time.

In this paper, we present two main contributions. First, we extended the tweet-collection functionality of NodeXL, a network visualization tool using Microsoft Excel, to allow aggregation of summary statistics (such as word and word pair frequencies) across multiple workbooks into a single workbook. Second, we developed a tool that visualizes the results of automatic topic detection and topic alignment between sets of tweets over time.

General Terms
information visualization, social networks, topic modeling

1. INTRODUCTION
With an ever increasing number of people using Twitter as a means of communication and live reporting during events, social scientists and researchers are beginning to use the analysis of Twitter data to gain insights about what users find interesting or important.

A variety of tools exist to visualize social media trends on a timeline [1, 10, 4, 13, 8, 9]. However, traditional analysis of Twitter data focuses on words, word pairs and hashtags. Although useful, this analysis is limited in that it does not provide a way to analyze complex topics that go beyond the scope of a user-defined hashtag or a word pair. To this end, we explored applying automatic topic modeling using the Latent Dirichlet Allocation (LDA) algorithm [2] to Twitter data in order to provide scaffolding for the temporal analysis of Twitter trends at a broader level. In order to allow for the analysis of changes in topics across time, our algorithm iteratively performs topic modeling on slices of Twitter data and algorithmically aligns the topics between time slices based on their similarity. Because we analyze the topics at discrete time slices separately, the topics of one time slice do not directly correspond to the topics of another. The issue of aligning topics is an open problem in Natural Language Processing (NLP), and we aim to solve it through our visualization tool, TopicFlow, which uses a Sankey [11] style graph to display how topics at discrete time slices are related.

In this paper, we discuss related works for Twitter data analysis and their limitations, present our solution for visualizing and aligning topics at discrete time slices, and evaluate the effectiveness of our tool with a panel of expert reviewers.

2. RELATED WORKS
Significant work has been done in the area of visualization of temporal Twitter information. The majority of this work involves the temporal analysis of hashtags, keywords, or tags. Conference Monitor [1] is a web-based Twitter visualization tool developed to study Twitter data surrounding a particular conference. This tool provides a useful visualization of hashtag usage over time. Similarly, Statler [13], developed by Yahoo! research labs to examine data as it corresponds to broadcast events, and TwitInfo [10] visualize keyword trends for Twitter data over time. Two tools, Nokia Internet Pulse [8] and SparkClouds [9], provide visualizations of hashtag cloud analysis. Nokia Internet Pulse provides a useful visualization of the evolution of a discussion surrounding a particular topic on Twitter with a time series of stacked tag clouds. Similarly, SparkClouds integrates sparklines into a tag cloud to convey trends between multiple tag clouds. Although useful, visualizations of this type limit analysis to trends represented by user-defined hashtags or specific keywords.

Diao et al. [5] perform a temporal topic analysis of microblogs using a topic modeling implementation designed to help in identifying “bursty” or “peaky” topics over time. Though interesting, and due to the explorative nature of their research, their method lacks a useful visualization of the data.

2.1 Background: LDA and Topic Alignment
Latent Dirichlet Allocation (LDA), a generative model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar [2]. LDA is one of the most common methods of statistical topic modeling, and was first presented as a graphical model for topic discovery in 2002 by David Blei et al. A topic model is a statistical model for discovering abstract concepts or “topics” that occur in a collection of documents. In LDA, each document is represented as a mixture of various topics, where each topic is a mixture of words. These mixtures are represented by P(topic|document) for all topics and documents and P(word|topic) for all words in the vocabulary. Each word may occur in several different topics with a different probability, and each document is assumed to be characterized by a particular set of topics. Typically, LDA algorithms use a list of stop words in order to exclude common and “noisy” words from the topic modeling.

Because topics are vectors of words and their probabilities, they can be compared by vector similarity metrics. Cosine similarity [16] is a measure of the similarity between two vectors that measures the cosine of the angle between them. This measure is regularly used for the comparison of documents or the cohesion of clusters in the fields of text mining and data mining, respectively. The cosine similarity measure puts higher weight on the more probable words in the topic’s word distribution, meaning two topics are considered to be more similar if they have higher probability words in common. This is useful for topic comparison, because the lower probability words in the distribution are typically representative of noise.

3. SOLUTION
3.1 Collecting Twitter data in NodeXL
NodeXL [15] is a free and open source Social Network Analysis Software created and designed by a group of people that mostly consists of university faculty and affiliates. It operates as a Microsoft Excel template. NodeXL produces various network visualizations (most of them based on a nodes and edges graph) that are highly customizable, as well as various network metrics.

It also allows for easy importing of specific types of network data, e.g. data that can be acquired through a Twitter API search. For example, during the the broadcast of the Presidential Debates, a user may choose to search for the hashtag “debate.” However, this functionality is limited to collecting only 1500 tweets at a time into a single workbook, and during times of high-volume tweets, these tweets could cover a very short time span. As a result, users will often create multiple workbooks at discrete timesteps to gather data over their desired time range. Users can then run “Aggregate Statistics Metrics” which compute the word and word pair frequencies from all the tweets in each workbook. This then allows the user to explore the tweet data at a single time slice, but does not allow the user to compare the word and word pair frequencies between workbooks (i.e., over a span of time). To enable temporal analysis over longer time ranges, we implemented a feature in NodeXL to consolidate the tweets and summary statistics from multiple workbooks into a single workbook.

3.1.1 Implementation

3.2 Analysis Pipeline

The TopicFlow analysis pipeline was developed in Python. The Python-based LDA implementation that was used to perform the statistical topic modeling of Twitter data was developed by Nakatini Shuyo [14]. The processed output was provided to the front end in the form of JSON files. The processing tool ingests NodeXL edge worksheets containing tweets that had been converted to the CSV format.

3.2.1 Binning the Corpus

The tweets are divided into some number of bins defined by an input parameter. Each bin represents a time slice of equal or almost equal length, therefore some bins contain more tweets than others (i.e. if the dataset contains more tweets of a specific time period).

3.2.2 Topic Modeling with LDA

Algorithm. The LDA algorithm is run for each bin separately, therefore producing a topic model for each bin where
each topic model contains some number of topics defined by an input parameter.

The result of topic modeling is typically a distribution of words for each topic, $P(\text{word} | \text{topic})$ in the topic model and a distribution of topics for each input document, $P(\text{topic} | \text{doc})$. For our use cases, we intended to provide the user with the ability to select a topic of interest and see all corresponding tweets. In order to enable this, we took for each tweet the highest $P(\text{topic} | \text{tweet})$ and assigned that tweet to the topic. Additionally, in presenting this information to the user, we rank the tweets by the probability, such that tweets that are mostly about the selected topics are shown first in the list, and tweets at the bottom have a more evenly distributed mixture of topics. There are many alternative methods that could have been employed to make this assignment, but we chose one that was straightforward and resulted in a good number of tweets being assigned to each topic.

**Design.** A parameter of the LDA algorithm is the number of topics. The current TopicFlow implementation does not expose this parameter to the user. For the initial development, we chose to set the number of topics to 15. We anticipated that this would be enough topics to enable the user to gain insights about the data, while not so many that they would overwhelm the user and the visualization.

The LDA algorithm uses a stop words list in order to remove common and noisy words from the topic modeling. In this implementation the stop words list contains the standard English stopwords. In order to account for Twitter data, the stop words list was modified to contain Twitter-specific stop words as well as Spanish stop words. The list also contains the words used in data collection for the individual datasets, due to the fact that the data was collected through keyword and hashtag search.

### 3.2.3 Aligning Topics

Similarity between the topics across time bins is calculated by using a cosine similarity metric. This metric is used to compare each pair of topics from time A to time B as vectors of the probabilities of the top 20 words in the topics. Cosine similarity is evaluated in such a way that topics that share high probability words are therefore considered more similar than topics that share low probability words - usually noise. Topic pairs are assigned a similarity value between 0 and 1, where 1 would represent the exact same topic. Although we could have used an algorithm to attempt to find the one most similar topic at time $n + 1$ for each topic at time $n$, we instead chose to show links for any topic pairs with similarity above a certain threshold. The goal being to enable the visualization of topic convergence and divergence. For this implementation, we set a threshold of 0.2, meaning that all topic pairs with similarity less than .2 were considered to be not similar. The remaining similarity values were weighted for best results in the visualization.

### 3.3 TopicFlow

TopicFlow was developed in Javascript, using jQuery [7] for the user interface and the Data-Driven Documents (d3) library [3] created by Mike Bostock for the visualization. We chose to use the d3 library because it is specifically designed for creating visualizations for large datasets, and contains many types of visualizations already implemented.

#### 3.3.1 Use Cases

TopicFlow was designed so that users of any skill level interested in a set of tweets corresponding to a specific event or time interval may use it. In order to import a data set into TopicFlow, the set of tweets must include each tweet’s author, timestamp, and text. The user may also pass in a set of stop words, typically including the word or hashtag that was used to search for and create the tweet set.

TopicFlow is intended to support six primary use cases.

1. Easily identify the most popular topics within each time slice. A topic is considered more popular if there are more tweets associated with it.

2. Easily identify which topics are emerging, ending, continuing, or standalone. Here we introduce four new terms:
   - **emerging** A topic that was not discussed in the previous time slice. These topics are considered to be topics that were introduced within its time slice because there are no topics similar to it in the previous time slice.
   - **ending** A topic that does not continue into the next time slice (i.e., there are no topic similar to it in the next time slice).
   - **continuing** A topic that has been discussed before and after its time slice.
   - **standalone** A topic that is not related to any topics in the previous or next time slice.

3. Explore details about a selected topic. These details include its most probably words, its most probably tweets, and the flow of a topic over time. The flow of a topic is considered to be the path of a topic and its related topics across all time slices.

4. Identify topics by the words that describe them. A user may be interested in how one or more words appear throughout the dataset. By identifying the topics that are related to these words, a user can understand how the context of a word changes throughout the dataset, as well as discover other words related to it.

5. Compare the top words in two topics that are related. By comparing two topics, a user can identify which words contributed to the topics having a high or low similarity score.

6. Filter topics and edges by size, type, and similarity. Users may want to view only highly similar or highly popular topics, and filtering will allow them to hide the topics in which they are not interested.

We chose to use an iterative development approach, guided by expert reviews. In each iteration, we 1) designed mock-ups to support our use cases and recommendations, 2) implemented these features into our visualization, and 3) conducted an external review to receive recommendations for our next iteration.
Figure 2: The first version of topic flow consisted of six coordinated windows: (1) the TopicFlow diagram, (2) a list of topics with their word summaries, (3) a space for topic details, (4) a list of the tweets in the dataset, (5) a space for tweet details, and (6) a filter pane.

4. ITERATIVE DESIGN AND EVALUATION PROCESS

4.1 Initial Version

4.1.1 Implementation
The initial version of TopicFlow consisted of six coordinated panels (Figure 2): (1) the TopicFlow diagram, (2) a list of topics with their word summaries (“topic list”), (3) a space for topic details (“topic details”), (4) a list of the tweets in the dataset (“tweet list”), (5) a space for tweet details (“tweet details”), and (6) a filter pane (“filters”).

1. TopicFlow diagram. In order to visualize the topic alignment over time, TopicFlow uses a novel visualization adapted from the d3 library’s Sankey diagram [11]. Our visualization displays the topics at each time range as nodes in the interface sized by the number of tweets attributed to the topic. They are ordered horizontally to minimize the number of edge crossings in the graph. The topics are colored by their type (emerging, continuing, ending, or standalone as described in Section 3.3.1). Edges between the nodes are weighted representations of the similarity between the topics, as calculated by the cosine similarity metric. The design of this visualization was motivated by Use Cases 1 and 2, and is successful in providing insights about the popularity and life-cycle of a topic within seconds of viewing it. When a user clicks an edge, a topic comparison box appears that shows which words the two topics have in common (Use Case 5).

2. Topic list. The topic list displays all the topics found in the dataset, grouped by their time slice. Each topic is named as time slice number_topic number, and displays a summary of the topic’s top ten words, sized by probability. The topic list also features a search box which allows users to find topics by a word they are interested in (Use Case 4).

3. Topic details. When a topic is selected from the topic list, this window shows a bar graph of the top ten most probable words for that topic (Use Case 3).

4. Tweet list. When no topic is selected, the tweet list displays every tweet from the dataset. Upon selecting a topic, the tweet list filters to contain only tweets related to that topic (Use Case 3).

5. Tweet details. When a tweet is selected, the window shows the full text of the tweet, a link to the author’s Twitter page, and a bar chart of the top five related topics to that tweet.

6. Filters. Users are able to filter nodes by size (number of tweets) or type (color) and edges by weight. This supports Use Case 6.

4.1.2 Evaluation
The first sessions were conducted with three participants, all of whom had used a Twitter data analysis visualization at least once before and have some graduate education or higher. The participants were recruited by email and word of mouth.

After a brief introduction and explanation of the tool, we allowed the participants to have a freeform exploration of the data (approximately 1500 tweets resulting from a search for the word “earthquake” over two days). They were instructed to describe everything that they were doing and why, as well as express any other comment that they might have (think-aloud method). Their comments and our observations (mistakes they made, unreasonable learning curves, bugs, confusing interface elements, missing items, etc.) were documented in handwritten and typed notes, taken by the researchers present during the session.

Results. Overall, the reviewers responded positively to the use of coordinated windows to display details-on-demand in each of the panes. The reviewers also appreciated the search box, which allows the user more control over filtering.

In addition to our positive feedback, the reviewers recommended a few changes that would augment our tool. The most requested features were:

R1 Coordinate TopicFlow diagram with topic list. When a topic is selected in the topic list, it should be selected in the TopicFlow diagram, and vice versa.

R2 Highlight search results from topic list in the TopicFlow diagram. Since the topic list is filtered upon search, reviewers felt the TopicFlow diagram should highlight this in some way.

R3 Show “subgraph” of a selected node in the visualization. When a node is selected in the diagram, all outgoing and incoming paths (related topics) should be highlighted.

R4 Allow users to select number of time slices and number of topics and add stop words to the LDA algorithm.

R5 Rank topics by size.
4.2 Final Version

In the next iteration, we were able to complete 4 out of the 5 recommendations from the usability review (R1, R2, R3, and R5). Due to time limitations, we were unable to bring LDA customizability to the frontend (R4), though this is something we hope to implement in the future. In addition to the recommended changes from reviewers, we also updated the look and feel of the layout to give a more consistent color palette and minimize unused screenspace.

4.2.1 Changes

- Coordinate TopicFlow diagram with topic list (R1). When a node is selected, either in the topic list or the visualization diagram, the corresponding node and list item are highlighted with a black border, as shown in Figure 3.

- Highlight search results from topic list in the TopicFlow diagram (R2). Upon search, the filtered topics are lightened so that the user can focus on the topics that match the search results.

- Show “subgraph” of a selected node in the visualization (R3). When a topic is selected in the visualization or topic list, nodes that have a path to that topic are highlighted, while unconnected topics are greyed out, as shown in Figure 4.

- Rank topics by size (R5). The nodes are ordered on the y-axis by decreasing size. This way, the most popular topics stay at the top and a user can also see how the popularity of a topic declines or increases over time.

- Move tweet and topic details to inside the tweet and topic lists respectively. In order to decrease clutter and maximize readability, we moved the details boxes to be embedded within their respective lists upon clicking (Figure 3). As a result, we have a cleaner, clutter-free 4-panel layout instead of 6.

- Use bar charts in topic comparison. The topic comparison window previous was just a list of words, and did not account for the weight of each word within a topic. Our new method involves displaying the bar charts for each topic, side by side, and highlighting the words they have in common (Figure 5).

4.2.2 Evaluation

In our second session, we conducted user reviews with two more participants. Both participants had some graduate education or higher and had used at least one Twitter data visualization tool at least once before.

Unlike our first session, our second session was more structured. We asked the participants to perform 7 specific tasks that exercised all the features of the visualization. For each task, the user rated the tasks on a 9-point Likert scale on efficiency (EI), correctness (C), effectiveness (EC), and satisfaction (S). The task list and questionnaire are available in Appendix A.

Results. Overall, results were positive. Across all tasks, the average metrics were as follows: efficiency - 7.4, correctness - 7.9, effectiveness - 7.4, and satisfaction - 7.9. In particular, reviewers felt the details on demand and brushing and linking were extremely useful for discovering insights about the datasets.

Recommendations included:

R1 Include more details about edge weights and topic similarities. Users noted that it would be useful if the tooltip on edges would show the value of the similarity metric, since it was at time difficult to determine the difference in weights between two edges. This could also be remedied by coloring the edges with a color scale.

R2 The topic comparison box needs more context. Users commented that the context of the two topics was lost when clicking on edges. A suggested fix was to highlight the edge and corresponding nodes when the topic box was fixed.

5. CONCLUSIONS AND FUTURE WORK

TopicFlow has been demonstrated as a useful tool for exploring Twitter data over time. Our interface employs the statistical nlp method of topic modeling to Twitter data, which allows for richer analysis of topic similarities within the data, beyond just single words or hashtags. When LDA is run over an entire corpus, it produces a high-level overview of the corpus content. Alternatively, TopicFlow splits the corpus into a set of time slices and runs LDA on each time slice. This method provides for a more granular set of topics, at the expense that topics must be aligned between time slices. A significant value of TopicFlow is that it provides a novel visualization of the alignment of topics over time. As topics are unnamed word distributions, providing a useful visualization of their similarity between time ranges is a valuable method for comparison.
Figure 3: TopicFlow contains four coordinated windows. Selecting a topic in either the topic list or the visualization will highlight the topic and its subgraph. Details for the selected topic and tweet are shown inline in their respective lists.

Figure 4: When searching for a word, the visualization will grey out the filtered topics as results are returned.
Evaluation demonstrated that users found our tool to be useful and interesting, and appreciated the interactions provided. We have identified a number of features and potential improvements to be incorporated in future implementations of TopicFlow.

5.1 Merging Workbooks in NodeXL
While the merge workbooks feature we implemented sufficiently served the purposes for our visualization, there are some improvements that could benefit it.

Firstly, the import process can become quite long when importing a large number of workbooks, and currently, there is no feedback about its progress. This may cause the users to become confused or think that the import has failed or frozen. A simple solution for this is adding a progress bar, that displays what percent of the import is complete. This would alleviate any doubts the user may have about the success of the import, as well as prevent the user from accidental clicks or interaction that may interfere with the import process.

Secondly, upon completion, the workbook should be checked for duplicates in edges and vertices. For edges, this would be a simple solution of deleting any duplicate edges. However, for the vertices, which represent a user at a particular point in time, it becomes more difficult. Suppose a user has a vertex created at time point A and again a later date, at time point B. The vertex contains information about the user’s number of followers, number of tweets, and other time-sensitive data. If any of this data changed between point A or point B, it becomes challenging to decide how to consolidate the user, if at all. One method would be to just use the most recent statistics. However, this will obscure any data about growth users may have had, especially growth as a result of something they tweeted. Another method could be to allow multiple vertices for a single user, but this could result in the network graph becoming crowded with vertices. In the future, we would like to research this problem further and find the optimal solution for consolidating duplicate vertices.

5.2 Analysis Pipeline
The current TopicFlow implementation employs a basic LDA implementation which runs over a set of text documents, representing the analysis and display of a larger variety of datasets, it will be important to allow the user to directly modify the stop words list. Additionally, stemming techniques have been shown to improve the results of LDA by retaining only the root of each word in the corpus, thus removing “duplicates” such as “earthquake” and “quake.” Integrating a stemmer into our pre-processing pipeline should lead to a higher quality topic model. Finally, the TopicFlow LDA implementation allows for the specification of two parameters, which are not currently exposed to the user. The number of bins and number of topics drive the topic modeling.

Another potential for improvement is in the topic modeling algorithm. Instead of running “vanilla” topic modeling on each bin of tweets, and then aligning the resulting topics, TopicFlow could make use of a dynamic topic model or a topics over time algorithm to topic model the tweets. In a future iteration, significant testing and evaluation will be necessary to determine which topic modeling algorithm produces the best results and is preferable by users performing temporal topic analysis of Twitter.

5.3 TopicFlow

5.3.1 Support user-uploaded data
Currently, TopicFlow provides the ability to analyze a static set of topic modeled twitter datasets. Typically a user will want to upload her own datasets for processing and visualization. The interface will be extended to allow the user to upload any dataset of text files to be processed and visualized, or, alternatively, to upload preprocessed data in the form of “binned” topic models.

5.3.2 Support comparison of multiple topics over time
Although evaluation showed that users found our visualization to be useful for analyzing particular topics for the Twitter corpora over time, some users noted that the ability to compare two topics would greatly added to the utility of the tool. In a future iteration, the TopicFlow team will experiment with different visualization techniques for best presenting this information to the user.

Additionally, in a next iteration of TopicFlow, we intend to modify the topic similarity panel to better support the comparison between topics from adjacent time slices. Specifically, based on user suggestion, this modification would include count or probability information for each of the words and alignment of the shared words within the bar graphs.

5.3.3 Provide additional topic layout options
In the initial implementation of TopicFlow, the topics were laid out in such a way to reduce edge crossings between time slices. Based on user suggestion, we instead switched to a top to bottom layout of the most prominent to least prominent topics at each time slice. This layout allows the user to easily focus on the topics with the desired level of importance. During implementation and evaluation we identified a number of potential layout options to explore in upcoming TopicFlow iterations, such as sorting by number of unique authors or unique hashtags for the tweets belonging to the topic.

6. ACKNOWLEDGEMENTS
We gratefully acknowledge the guidance of Ben Shneiderman, Cody Dunne, Marc Smith, Jana Diesner, John Alexis Guerra Gómez, Jordan Boyd-Graber, Jen Golbeck, Megan Monroe, Timothy Hawes, and Awalin Sopan for their valuable help during the multiple stages of design, evaluation, and implementation of our tools.

7. CREDITS
Jianyu Li contributed to design and implementation of aggregating multiple workbooks in NodeXL, participated in brainstorming ideas for TopicFlow and participated
in the initial mockups design, implemented many functions of the backend of TopicFlow, assisted in conducting the evaluation session, outlined the draft paper, as well as participated in the writing and editing of the final paper and presentation.

**Sana Malik** contributed to designing and implementing the import dialog for the NodeXL merge workbook feature, brainstorming and designing mockups for TopicFlow, implementing various features in the frontend of the visualization (live search, brushing and linking, filtering), conducting the usability sessions, and writing, editing, and formatting of the final paper and presentation.

**Pano Papadatos** contributed to designing initial mockups for TopicFlow as well as in the brainstorming and discussion concerning the visualization and implemented many functions related to the processing of the data before it is served to the visualization tool. He contributed in the design of the evaluation sessions and in conducting them. Finally, he wrote the video script and produced the video, as well as participated in the writing and editing of the final paper and presentation.

**Alison Smith** contributed to designing mockups for the TopicFlow interface and visualization, modified and integrated with the LDA algorithm and implemented topic alignment in the analysis pipeline, implemented various features in the interface (topic similarity tooltip, initial layout and coordinated panel functionality), assisted with the overall design of the visualization, and participated in the writing and editing of the final paper and presentation.

### 8. REFERENCES


APPENDIX
A. USER STUDY DOCUMENTS
A.1 Tasks

- Identify the two most similar topics
- Identify the reason why these two topics are similar
- Identify a topic that did not die out for this timespan
- Identify a topic that emerged and died really soon
- Identify a topic that diverged into more than one topic
- Pick a topic and find out what it is about in depth
- Identify 3 of the most important words

A.2 Questionnaire

1. How efficient was TopicFlow in aiding you achieve the task you just attempted on a scale from 1 to 9, 1 being very inefficient and 9 being very efficient?

2. How correct were the results for the task you just attempted on a scale from 1 to 9, 1 being very incorrect and 9 being very correct? (i.e., how much do you trust your conclusions)

3. How effective was TopicFlow in aiding you achieve the task you just attempted on a scale from 1 to 9, 1 being very ineffective and 9 being very effective?

4. How satisfied are you in regards to this task on a scale from 1 to 9, 1 being very unsatisfied and 9 being very satisfied?