Event Trend Detector
Final Report
CS4624 - Multimedia, Hypertext, and Information Access
Skylar Edwards, Ryan Ward, Stuart Beard, Spencer Su, Junho Lee

Client: Liuqing Li
Instructor: Edward A. Fox
May 7, 2018
Virginia Tech, Blacksburg VA 24061
# Table of Contents

Table of Figures ........................................................................................................ 5

Executive Summary ................................................................................................... 6

1 Introduction ............................................................................................................. 7

2 Requirements .......................................................................................................... 8
   2.1 Project Deliverables .......................................................................................... 8
       2.1.1 Clustering .................................................................................................. 8
       2.1.2 User Interface ............................................................................................ 8
       2.1.3 Event Trend Detection ..............................................................................

3 Design ..................................................................................................................... 10
   3.1 Clustering ......................................................................................................... 10
   3.2 User Interface .................................................................................................. 10
       3.2.1 Trend Graphs ............................................................................................ 10
       3.2.2 Carousel View ........................................................................................ 10
       3.2.1 Initial Mockup Design ............................................................................. 11
       3.2.2 Final Design ............................................................................................. 11
   3.3 Trend Detection ............................................................................................... 13
       3.3.1 Trend Detection Flow ............................................................................... 13
       3.3.2 Trend Table ............................................................................................... 13
           3.3.2.1 Initial Design .................................................................................... 14
           3.3.2.2 Intermediate Design ....................................................................... 14
           3.3.2.3 Final Design ....................................................................................... 15
       3.3.3 Tagged Entities ........................................................................................ 15

4 Implementation ....................................................................................................... 16
   4.1 Data Extraction ............................................................................................... 16
   4.2 Data Processing ............................................................................................... 16
   4.3 Server and Database ....................................................................................... 18
       4.4 Structure ....................................................................................................... 18
       4.5 Version Control ............................................................................................ 19

5 Testing/Evaluation/Assessment .............................................................................. 20
5.1 Data Extraction Testing

5.1.1 Poller Testing

5.2 Data Processing Testing

5.2.1 SNER Testing

5.3 Cluster Testing

5.4 Website Usability Testing

5.5 Database Connection Test

6 Future Work

6.1 Cluster Filtering

6.2 Domain Authority Rank

6.3 Trend Detection Query

6.4 Additional Sources

7 User Manual

7.1 Navigation

7.1.2 Clustering

7.1.3 Trends

7.2 User Roles

8 Developer’s Manual

8.1 Databases

8.2 Back-End Code

8.2.1 Control Flow Between Files

8.2.2 poller.py

8.2.3 article.py

8.2.4 articleCluster.py

8.2.5 processNews.py

8.2.6 driver.sh

8.2.7 populateTable.py

8.2.8 google-trends.py

8.2.9 reddit-trends.py

8.3 Front-end Trend Display code

8.3.1 .htaccess

8.3.2 config.php

8.3.3 global.php

8.3.4 siteController.php
8.3.5 home.tpl ................................................................. 35
8.3.6 public/ ................................................................. 35
8.4 Cluster Display Code ................................................... 35
  8.4.1 ball_animation_1.js ............................................... 35
  8.4.2 cluster.php ........................................................ 36
  8.4.3 index.php .......................................................... 36

9 Lessons Learned ................................................................ 36
  9.1 Use Existing Tools ..................................................... 36
  9.2 Start Early ............................................................... 36
  9.3 Research ................................................................. 36
  9.4 Regularly Scheduled Meetings ..................................... 37
  9.5 Documentation ........................................................ 37

Acknowledgments ................................................................ 38

References ........................................................................... 39

Appendices .......................................................................... 40
  Appendix A Milestones and Timeline .................................. 40
    A.1 February ............................................................... 40
    A.2 March ................................................................. 40
      A.2.1 Milestone 3 (02/23 to 03/09): .............................. 40
      A.2.2 Milestone 4 (03/09 to 03/23): .............................. 40
    A.3 April ................................................................. 41
      A.3.1 Milestone 5 (03/23 to 04/06): .............................. 41
      A.3.2 Milestone 6 (04/06 to 04/20): .............................. 41
      A.3.3 Milestone 7 (04/20 to 05/02): .............................. 41
  Appendix B Completed Work ............................................. 42
  Appendix C Table of Routines ............................................ 44
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Initial Trend Graph</td>
<td>11</td>
</tr>
<tr>
<td>3.2</td>
<td>Final Trend Graph</td>
<td>12</td>
</tr>
<tr>
<td>3.3</td>
<td>Final Cluster UI Design</td>
<td>12</td>
</tr>
<tr>
<td>3.4</td>
<td>Trend Detection Flow</td>
<td>13</td>
</tr>
<tr>
<td>3.5</td>
<td>Initial Trend Table Design</td>
<td>14</td>
</tr>
<tr>
<td>3.6</td>
<td>Intermediate Trend Table</td>
<td>15</td>
</tr>
<tr>
<td>4.1</td>
<td>Raw Content Array</td>
<td>17</td>
</tr>
<tr>
<td>4.2</td>
<td>Article Title and Content with Stopwords Filtered Out</td>
<td>17</td>
</tr>
<tr>
<td>4.3</td>
<td>Example of Extraneous Content</td>
<td>17</td>
</tr>
<tr>
<td>4.4</td>
<td>Filtered Content</td>
<td>18</td>
</tr>
<tr>
<td>4.5</td>
<td>Overall Control Flow of Current Setup</td>
<td>18</td>
</tr>
<tr>
<td>8.1</td>
<td>Data Flow between Files</td>
<td>26</td>
</tr>
<tr>
<td>8.2</td>
<td>Raw Content Array</td>
<td>27</td>
</tr>
<tr>
<td>8.3</td>
<td>Data Flow within processNews.py</td>
<td>28</td>
</tr>
<tr>
<td>8.4</td>
<td>Results from Clustering Function</td>
<td>31</td>
</tr>
<tr>
<td>8.5</td>
<td>Results from SNER</td>
<td>32</td>
</tr>
<tr>
<td>8.6</td>
<td>Cleaned SNER Data</td>
<td>33</td>
</tr>
<tr>
<td>8.7</td>
<td>Tagged Entity Database Table</td>
<td>33</td>
</tr>
</tbody>
</table>
Executive Summary

The Global Event and Trend Archive Research (GETAR) project is supported by NSF (IIS-1619028 and 1619371) through 2019. It will devise interactive, integrated, digital library/archive systems coupled with linked and expert-curated webpage/tweet collections. In support of GETAR, the 2017 project built a tool to scrape the news to identify important global events. It generates seeds (URLs of relevant webpages, as well as Twitter-related hashtags and keywords and mentions). A display of the results can be seen from the hall outside 2030 Torgersen Hall.

This project extends that work in multiple ways. The quality of the work done has been improved. This is evident in changes done to the clustering algorithm and the user interface changes to the clustering display of global events. Second, in addition to events reported in the news, trends have been identified, and a database of trends and related events were built with a corresponding user interface according to the client’s preferences. Third, the results of the detection are connected to software for collecting tweets and crawling webpages, so automated daily runs find and archive webpages related to each trend and event.

The final deliverables include development of a trend detection feature with Reddit news, integration of Google Trends into trend detection, an improved clustering algorithm to have more accurate clusters according to k-means, an improved UI for important global events according to what the client wanted, and an aesthetically pleasing UI to display the trend information. Work accomplished included setting up a table of tagged entities for trend detection and configuring the database for clustering and trends to work with our personal machines, and completing the deliverables. Many lessons were learned regarding the importance of using existing tools, starting early, doing research, having regular meetings, and having good documentation.
1 Introduction

The main goal of the Event Trend Detector is to build upon the work of the previous group’s Global Event Detector in certain ways [1]. The main improvement required by the client is to detect and visualize news trends that are prevalently showing up on news sites within a 3 day span. We were given the additional tasks of improving the clustering algorithm and improving the UI of the previous group’s event detector as well.

The trend detection component takes the database of the previous group and identifies high frequency news stories and shows the data using a line graph of how often the keyword occurs in Reddit news stories as well as sorting it by name, organization, and location.

The clustering improvement consists of changing the clustering algorithm to best represent k-means. We devised a method of identifying the datapoint that best represents a cluster and tested various thresholds of similar keywords to create the clusters. The dataset that the clustering is performed on contains processed articles that have been represented as vectors.

All the data is scraped from Reddit news as well as Google Trends [5, 11]. The results of the clustering and the trend data will be shown on a screen. The UI displays the trends through a line graph and the clusters are shown using bubble graphs that represent each cluster created as a bubble on the screen. The cluster and the trend screens will automatically be scrolled to display them for an interval of at least 15 seconds.
2 Requirements

2.1 Project Deliverables

The client for the Event Trend Detection group gave three distinct deliverables at the beginning of the project. The first deliverable was to improve the clustering algorithm from the previous group’s work. The second deliverable that the client gave us was to improve the current user interface. The last deliverable was to implement trend detection and have it displayed within the user interface.

2.1.1 Clustering

The first requirement was to improve the clustering algorithm used by the previous group. We were tasked with improving the efficiency of the algorithm and detailing the results. The client wants to see how the changes to the algorithm affect the current status of the project.

2.1.2 User Interface

The next main requirement for the project was to improve the user interface. The client wanted a clean and easy to read user interface to visualize the trend detection data and article clustering data.

The user interface is not able to be interacted with because it is displayed behind a window on a computer monitor. Due to this, the client gave the requirement that the application can be fully utilized without physical interaction. The display is automatic and timed so a user can get a full summary within a few minutes.

2.1.3 Event Trend Detection

The last main requirement given to the group by the client was to implement a way to detect trends on Reddit and other news sources such as Google Trends. The trends would later need to be displayed within the user interface in a visual format.

In addition to the detection and implementation, another requirement for the trends was to properly store the data in a MySQL database. The database and server were provided by the client for this specific requirement.
The client defined a trend as a specific event, location or organization that is currently in the news or being talked about over a certain time period.
3 Design

3.1 Clustering

The previous group designed the clustering to take in the tokenized, processed news articles and assigned an ID to each word in a dictionary they created and measured the frequency of all the words. They used a TF-IDF weighting for the words and then created a vector for each article.

To build off of this initial design, we decided to implement a k-means method of clustering instead of just identifying the threshold of similarity between each document. This would help better create clusters that have better representation points. We also adjusted the clustering algorithm to take in more than just Reddit IDs to account for the additional news sources that the scraper will read from. We created cliques from graph theory to link close articles together and took the mean of the clique to represent each of the clusters.

3.2 User Interface

3.2.1 Trend Graphs

In order to improve the user interface, we needed to think about how typical trends would be displayed. In order to represent a trend in a visual format, we decided that each trend should have its own graph indicating the frequency of occurrence over a certain time interval. In each graph in the user interface we used the x axis to represent time and the y axis to represent the frequency of the trend. Trends are detected based on the most frequent mentions of named entities over the past week, and the graph of each trending entity’s mentions over time is added to a carousel of trend graphs. The graph of each trending entity’s mentions covers as long a time period as we have data for.

3.2.2 Carousel View

In addition to creating visual representations of the trend graphs we needed to make sure that the data was automatically cycled through. For example, the trend graphs would need to be displayed within a carousel view that automatically changed after fifteen seconds. The automatic carousel view offered the perfect interface for the trend graphs because it requires no interaction from the user.
3.2.1 Initial Mockup Design

Figure 3.1 shows a simple mockup of the initial design for the user interface. There is a trend graph that automatically cycles after a certain period of time. Problems with this design include issues with labeling and allocation of screen space. We have two horizontal monitors to display data on, with trends and clustering being the important parts. The mockup design below is vertical and has no space for clusters to be displayed.

![Initial Trend Graph Design](image)

3.2.2 Final Design

The final design (Figures 3.2 and 3.3) switches from the original mockup (Figure 3.1) to fit a horizontal monitor. The top bar is removed and the indication of “today’s” trends and articles is switched to “current”. Space is used more efficiently in order to display all the relevant information. Clusters are displayed on a separate monitor. The clusters are displayed in the
form of moving bubbles that slowly move around on screen according to the request of the client. Trend graphs automatically rotate between the current detected trends and display all the trend data for these topics from the past year. The news carousel from the previous group is displayed with the trends as well.

Figure 3.2 Final Trend Graph Design

Global Event Trend Detection

- A Syria decision hadn't been made when Trump tweeted missiles will be coming
- Mexican Drug Cartels Warn Politicians Drop Out or Be Killed As Presidential Election Nears
- Israeli intelligence reportedly says Trump's Syria strike failed, didn't take out much of anything
- A sperm whale that washed up on a beach in Spain had 64 pounds of plastic and waste in its stomach
- International chemical weapons watchdog confirms UK analysis of nerve agent used in Russian ex-spy poisoning
- North and South Korea reportedly set to announce official end to war
- Facebook's tracking of non-users sparks broader privacy concerns – Zuckerberg said that, for security reasons, the company collects data of people who have not signed up for Facebook

Figure 3.3 Final Cluster UI Design
3.3 Trend Detection

3.3.1 Trend Detection Flow

In the design of the trend detection component, we created a simple flow chart to illustrate the process behind creating the trends. The previous group created a list of tagged entities for each article, so we designed the trend detection to count the frequencies of these tagged entities over time. The most mentioned entities of the past week are labelled as trends. Below is the flow chart to show the trend detection design.

![Trend Detection Flow Chart](image)

3.3.2 Trend Table

When designing the trend table for the MySQL database we needed to think about how the trend was going to be displayed in the user interface. During our initial research phase, we used websites such as Google Trends to determine how trends are displayed[11]. Using the information we gathered during the research phase we were able to design a trend table with key characteristics.
3.3.2.1 Initial Design

The initial design of the trend table included five distinct columns. The columns in the table included a unique ID, a name, a date, a frequency and a URL. The unique ID acted as an identifier for each trend. The name served as a description for the trend such as “Trump” or “Russia”. The date served as an identifier for whether the trend was ongoing, and if it was not, when it stopped being a trend. The frequency was a counter of the number of occurrences for the trend. The last item was a URL which would link the trend to a specific article. Below is a diagram of the initial design.

![Initial Trend Table Design](image)

3.3.2.2 Intermediate Design

The intermediate design of the trend table was an improved version after discussing with our client. In order to properly display a trend we needed to have a time interval over which the trend occurred. Therefore in our intermediate design for the trend table we created a start date and end date column. Additionally we created a Boolean to
determine if a trend was currently trending or if it was not trending. Next we decided to add a tag field to be able to identify the trend type by location, person or organization. Lastly we removed the URL field because we felt that it was not necessary to properly identify a trend.

<table>
<thead>
<tr>
<th>Trend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Key</td>
</tr>
<tr>
<td>id</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>frequency</td>
</tr>
<tr>
<td>start_date</td>
</tr>
<tr>
<td>end_date</td>
</tr>
<tr>
<td>isTrending</td>
</tr>
<tr>
<td>tag</td>
</tr>
</tbody>
</table>

Figure 3.6 Intermediate Trend Table Design

3.3.2.3 Final Design
The final design uses only the tagged entities database table to keep track of what entities have been tagged in news articles from the entire period of time we have been gathering data. We keep the name, type/tag (Person, Organization, Location), and the date the entity was tagged on. To detect a trend, we find which entities have been tagged an above average number of times over the past week and label those as trends. Any trends have graphs displayed about the number of times the respective entity was tagged during each month over the past year, for Google Trends and Reddit separately [5, 11].

3.3.3 Tagged Entities
The previous group used a natural language processing library called SNER to identify tagged entities associated with each article[4]. The tagged entities were sorted by
person, location and organization. The design for the trend detection was to take these tagged entities and count how often they appear over a certain time interval to generate the trends. For example if “Russia” was tagged as a location from December 31st to January 7th 100 total times then we would create a new trend called “Russia”. The trend would have a frequency of 100 and a time interval from the 31st of December to January 7th. We could then take this trend and display its information in a graph within the user interface.

4 Implementation

To implement this project we needed to use a variety of existing libraries and tools. The purposes of these libraries and how they fit into the larger project structure will be outlined here.

4.1 Data Extraction

The web scraping portion of this project was done using Python 3. In addition to using the many default libraries that Python provides, several key libraries were needed. PRAW (Python Reddit API Wrapper) was used to get the top daily posts off of the WorldNews subreddit [5]. Urlib is used to send the GET requests for the raw HTML files from URLs and Gensim and BeautifulSoup4 are used to extract the article information out of the HTML files in an attempt to eliminate any extraneous information which is unrelated to the article[13]. The PyMySQL library was used to take the parsed data and store it in a SQL database.

4.2 Data Processing

Once the raw data has been extracted and placed in the database it must be processed. The data processing portion of the project was also implemented using Python; several different libraries were needed. The NLTK (Natural Language Toolkit) was used for natural language processing so that articles could be condensed into a series of key words after removing stopwords [3]. This is shown in Figures 4.1 through 4.4. The SNER (Stanford Named Entity Recogniser) was used to determine which words are named entities, such as the names of people and places, and place them into categories [4]. NetworkX is used to create a graph data structure of related news stories [2]. Once the data is processed it gets placed into a different table from the raw data using PyMySQL.
Figure 4.1: Raw Content Array [1]

Figure 4.2: Article Title and Content with Stopwords Filtered Out [1]

Figure 4.3: Example of Extraneous Content [1]
4.3 Server and Database

The server was created using XAMPP, a cross platform Apache web-service tool[15]. XAMPP provides an easy user interface to set up an Apache server and a MySQL database. XAMPP also includes phpMyAdmin, a database tool for easy creation of MySQL databases[15, 16]. This was used to create the databases used for trend and event storage.

4.4 Structure
4.5 Version Control

We have a private Github repository set up for version control and for backup purposes. This github is not meant for future GETAR projects to use as the repository was only created for personal use within the scope of this project.
5 Testing/Evaluation/Assessment

5.1 Data Extraction Testing
Data extraction was done through utilization of PRAW API, which collected articles from Reddit pages [5]. In order to test some special cases, we had to handle those in testing.

5.1.1 Poller Testing
Not all of the domain sources allowed articles to be accessed by PRAW API requests and they returned an HTTP 403 forbidden error. Due to the unavailability of the article data, it was essential to investigate the reason for such access failure through testing. Some of the URLs returned an HTTP 404 page not found error when using the URLs from PRAW API to collect the HTML content of articles for parsing. The URL requests that could possibly result in errors were put into Python Try-Except statements which logged error messages for website links and moved on to the next URL, skipping the URL that caused the error. This ensured the articles polled from Reddit were functional and ready to be processed with the natural language techniques, and then clustered.

5.2 Data Processing Testing
Data processing was done in two major ways: NLTK tokenization, and SNER tagging. The NLTK tokenization functionality was proved to be credible after showing satisfactory and consistent tokenization on several runs.

5.2.1 SNER Testing
Testing SNER output was done manually by matching the human-judged relevance of the tagged outputs of several different pre-trained models. The pre-trained models identified three to seven classes of words. For example, the model which identified three different class models identified location, person, and organization. The words that were not identified were listed as ‘other’ and subsequently discarded because they would serve no purpose in clustering nor trend detection. The seven class model was able to successfully identify money, percent, date, and time in addition. After inspecting the tagging results of the SNER library, the group came to the conclusion that the sample datasets would be best used if the three class model was implemented, since more detailed tagging was increasing the tagging-noise (tagging that has no crucial meaning) within the tagged result. Due to the larger volume training set of this project, the group decided to minimize such noise that would bring unconcentrated result in clustering or trend detection.
5.3 Cluster Testing

When testing the output of something as complex as Clustering, the easiest thing to do is to start working with a dataset where the clusters are clearly visible with the human eye, and then slowly work with more complex datasets. This incremental testing is how we tested our clustering algorithm. We kept a baseline of similarity that we adjusted to see what effect it had on the resulting clusters until we found better results than what was implemented previously.

5.4 Website Usability Testing

We showed the website to different people and requested feedback on various aspects, such as ease of use or whether or not the time spent on each trend in the automatic changing display was too long or too short. Additionally, every update made to the website had been discussed with the client and implemented according to the client’s feedback and approval.

5.5 Database Connection Test

Using a Python Script called dbtest.py we test whether or not a successful connection can be made to the created databases. The script simply attempts to make a connection and if there is a failure outputs the error trace.
6 Future Work

6.1 Cluster Filtering

Currently, the data visualization is presented in non-filtered manner, meaning that all gathered data in the backend system will be displayed. In clustering visualization, all data clusters will be shown as separate clusters, but in one visualization. This may lower the accuracy of the clustering - for instance, a data cluster for “Trump” may contain both data about Donald Trump diplomatic policies and internal-economic policies. Clearly in this case, the clustering topic “Trump” may be too broad for one cluster visualization. Through cluster filtering, clusters may provide more accurate visualization of data, and better follow human perception of the presented data. Cluster filtering may contain options not limited to topics, but including time, region, source of data, persons, and organizations.

6.2 Domain Authority Rank

The purpose of domain rank is to determine how reliable and popular news article sources are. We could use Mozscape API to evaluate the authority of a given data/article. Through Mozscape, it is possible to predict how data would rank on a search engine, which could be one criteria for measuring the Authority Rank. The future work is to integrate the Mozscape prediction/evaluation functionality with the clustering to effectively visualize the credibility of the sources for clustering.

6.3 Trend Detection Query

The purpose of trend detection is to measure a certain topic appearance frequency and present the organized frequency information in visualized format. The current trend detection model includes a defined topic for trend detection, due to lack of other input methods by a user. The display is located at Torgersen Hall 2030, without any input device. For such reason, searching for a specific topic trend is a unavailable option for trend detection functionality of this project. For future implementation, for better user experience, it may be permitted that the user can provide an input for the trend detection functionality, to search for the trend of a specific topic. Additionally, we could have a server that supports multiple clients, e.g., one with an automatic display and others issuing queries.
6.4 Additional Sources

The final source of clustering and trend detection consists of articles collected from Reddit using the PRAW API and Google Trends. In order to provide more variety and dimension in data analysis of this project it is optimal to include other different sources of data as well. In order to achieve the diversity, a new source of data could be introduced in the future. Google News supports more than 30,000 different news-media sources that have high quality contents with well-known validity. Thus, data extraction will now be done not only relying to Reddit PRAW API and Google Trends, but also Google News API [12], and any other news collection site.
7 User Manual

The following sections will explain what future users of the event trend detector will see and how they will interact with each part of the project.

7.1 Navigation

The user interface is non-interactive because it is displayed on a monitor behind a glass window in Torgersen Hall. Without user interaction, the trends and clustered articles need to automatically cycle through. The user interface must be automatic in design.

7.1.2 Clustering

The user will observe the clusters created by the Event Trend Detector in the form of a bubble graph, where every bubble will have a representative article identifying what the articles in the clusters are talking about. The bubbles are similarly scaled no matter the number of articles belonging to each cluster as the amount of articles for each cluster over a three day span doesn’t vary enough to warrant that as a feature. The user will see that bubbles bounce off each other for an enhanced visual aesthetic.

7.1.3 Trends

The user will be able to visually interact with the trends by viewing the related graphs. Each graph consists of a different trend name, type, frequency and time interval. The trend graphs would be displayed to the user similarly to the article clusters. In order to maintain the automatic display, the trend graphs will be set on a carousel to cycle through for the users. The users can see the trend in a visual manner and come up with their own conclusions.

7.2 User Roles

There are many different types of users that will be looking at the event trend detector.

- Users that are looking for news information
  - Users interested in trending news topics: Individuals who are interested in what news stories are continuously being discussed in a three-day span.
  - Users interested in popular news now: Individuals that want to see what news topics people are discussing the most.
  - Browsing individuals who are curious: People who are just walking by Torgersen 2030 and are curious about what the project does.
- Researchers
- Researchers involved with GETAR: Members of Virginia Tech faculty involved with GETAR that will use the additional features of the trend detector to move on with their project
- News Researchers: Researchers that are interested in how news trends and how to identify incoming trends.

- Students
- CS 4624 students: Any future students of this course that will be given the task of improving upon the GETAR project.
8 Developer’s Manual

The following section contains information for future developers working on the project, much of which is passed down from the previous project group with our client’s permission.

8.1 Databases

There are four database tables to store data used in this project: [1]

1. The raw database table stores values from the Subreddit object obtained from the PRAW API. Values stored include RedditID, URL, title, content, date posted, date accessed, number of comments, number of votes, and domain name. The content is retrieved directly after processing an article’s HTML using BeautifulSoup.

2. The processed table stores the data from the raw database table after text processing, clustering, and seed extraction have been applied to the raw data. Values stored include process ID, processed date, processed title, seeds, and article score.

3. The clusterPast table stores a historical account of all the cluster data over time. Values stored include cluster ID, cluster array, and cluster size. The cluster ID is for the representative article for the cluster, the cluster array shows which articles are most similar to each other, and the cluster size records how big each cluster is.

4. The cluster table stores the same information as the clusterPast table but only maintains the results of the clustering algorithm for a single run. It is used for visualizations to display the size and content of similar articles.

5. The Tagged_Entities table stores information about entities identified in news articles, including the name of the entities, the dates they were tagged, and the types of entities they are. This information is used to identify trends and display information about them.

8.2 Back-End Code

Back-end code is written in Python and Bash and is responsible for processing the data that drives the webpage.

8.2.1 Control Flow Between Files

Polling, text processing, and data analytics are performed using three main files: driver.sh, poller.py, and processNews.py. Article.py and articleCluster.py are wrapper files for objects. Figure 8.1 is a graphical representation of data flow between files.
8.2.2 poller.py

Poller.py is responsible for scraping the ten “hottest” links off the WorldNews subreddit and storing the information gathered into the raw database. The PRAW API is used to grab the top ten links and the URLs are parsed with BeautifulSoup to grab content and parse out extraneous HTML like navigation bars. This content is called raw content and is stored in the raw database. Figure 8.2 shows an example raw content array based on the previous group’s design.
8.2.3 article.py

This file defines a NewsArticle object, which contains information about a news article. Fields of the NewsArticle object are populated from the raw database, including:

- **URL**: string version of a URL scraped from Reddit
- **title**: string version of the title scraped from Reddit
- **redditID**: string version of the RedditID for an article
- **rawContent**: word tokenized version of the raw content retrieved from parsing HTML, as described in section 8.2.2

Other fields are computed from the components listed above. These fields will be stored in the processed database:

- **content**: Processed content. Processed raw content is word-tokenized and stored in this field.
- **cluster**: List of clusters that this article belongs to. Also a RedditID.
- **entities**: List of all named entities in an article
- **taggedEntities**: List of the most popular named entities with associated tag. Tag can be a person, organization, or location.

Other fields are used to help with processing but are not stored in a database. These fields are:

- **procTitle**: Word-tokenized processed title field. This title field goes through the same processing stages as the content field.
- **simArts**: List of articles, identified by RedditID, that this article is similar to
8.2.4 articleCluster.py
This file contains a definition for a Cluster object. This information is stored in the two cluster databases. Components include:
- redditID: The RedditID of the representative article which is used to ID the cluster as a whole.
- articles: List of NewsArticle objects in this cluster

8.2.5 processNews.py
This file parses article content, clusters articles, and extracts seeds from article content. Figure 8.3 shows the data flow for the text processing functions within processNews.py.

Figure 8.3: Data Flow within processNews.py [1]

Two global arrays hold all NewsArticle and Cluster objects.

processNews.py extracts information from the raw database table, word-tokenizes it, and stores it into a NewsArticle object. The raw content has stopwords removed with the help of NLTK from both the content and title, and the resulting list of words is stored appropriately into the content and procTitle fields. Extraneous HTML content like suggested stories and comments is removed by comparing words in the content to words in the title using the GoogleNews pre-trained word vector model [7]. Words that are over 40 percent similar are kept, as the previous group’s testing showed that this threshold produced the best results [1]. The threshold starts from 100 percent, and decreases to find the best threshold, while the bottom-line threshold is 40 percent.
Words are then stemmed using NLTK’s Porter Stemmer [8] so that word endings and capitalization do not affect results. Seeds are then extracted using the NLTK part of speech tagger [9]. The full list of named entities is stored in the NewsArticles objects. Entities are tagged using the Stanford Named Entity Recognizer. Location, Person, and Organization are identified and unidentified words are listed as Other, labeled as ‘O’ in the Python object, and discarded. Multi-word entities are tagged as a whole and word-by-word. For example, “Donald Trump” and “Donald” and “Trump”. The 5 most frequent locations, people, and organizations are stored as tagged entities in the article object [1].

Next articles are clustered. The clustering function takes in word tokenized, processed news article content and finds the frequency of each word in the article. A graph is then created where each node represents a NewsArticle object. Articles which are at least 15 percent similar to each other have an edge drawn between them. The Python NetworkX library [10] is used to find cliques, and then subsequently clusters. Each cluster is represented by a randomly chosen representative article’s RedditID and each NewsArticle object stores a list of clusters it is a part of. Figure 8.4 shows the output of clustering on a full week of data from around April of 2017.

<table>
<thead>
<tr>
<th>Cluster: 671pt</th>
</tr>
</thead>
<tbody>
<tr>
<td>North Korea says ready to strike U.S. aircraft carrier</td>
</tr>
<tr>
<td>North Korea 'will test missiles weekly', senior official tells BCC</td>
</tr>
<tr>
<td>North Korea: 'US has now gone seriously mad' -- US strike group to arrive off Korean peninsula in days amid concerns the North is ramping up for a sixth nuclear test.</td>
</tr>
<tr>
<td>US military considers shooting down North Korea missile tests</td>
</tr>
<tr>
<td>Putin sends troops to Russia's border with North Korea</td>
</tr>
<tr>
<td>Chances of imminent war with North Korea 'wildly overblown', U.S. experts say</td>
</tr>
<tr>
<td>North Korea warns Australia of 'blindly toeing US line', warns of nuclear strike</td>
</tr>
<tr>
<td>Kim Jong-un is starting to get 'very paranoid', UN ambassador warns</td>
</tr>
<tr>
<td>South Korea tells Trump It's Actually Never Been a Part of China</td>
</tr>
<tr>
<td>North Korea warns it will 'wipe America off face of the Earth' after accusing US of plotting chemical weapons attack</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cluster: 674kth</th>
</tr>
</thead>
<tbody>
<tr>
<td>France’s politicians are uniting against far-right candidate Marine Le Pen</td>
</tr>
<tr>
<td>Macron-Le Pen ‘in French run-off’</td>
</tr>
<tr>
<td>Melenchon: Far-leftist surges in French polls, shocking the frontrunners</td>
</tr>
<tr>
<td>Defeated conservative Fillon calls on supporters to choose Macron over Le Pen</td>
</tr>
<tr>
<td>French mayor threatens to quit after town votes for Le Pen</td>
</tr>
<tr>
<td>Russia-linked fake news floods French social media</td>
</tr>
</tbody>
</table>
Cluster: 65us56
EU observer in Turkey condemns referendum as 'neither fair nor free'
"Intimidation and unfair campaigning" put Turkey referendum 'below international standards'
Hundreds protest against Turkish referendum result
Trump called Erdogan to Congratulate Him on Referendum Results, Sources Say
Up to 2.5 million votes could have been manipulated in Sunday's Turkish referendum that ended in a close "yes" vote for greater presidential powers, an Austrian member of the Council of Europe observes

Cluster: 65jzw6
Unsubsidized wind and solar now the cheapest source for new electric power
Britain powered 24 hours without coal for first time in 135 years in 'watershed moment'
Britain sets first coal-free day since the industrial revolution: The UK is set to have its first ever working day without coal power generation since the industrial revolution on Friday, according

Cluster: 66zhb6
Sources: US prepares charges against Wikileaks' Assange
Wikileaks releases more top-secret CIA docs as U.S. considers charges: Wikileaks claims the 31-page user guide for a CIA device code-named "Weeping Angel" can turn some Samsung TVs into surveillance t

Cluster: 6632jd
Saudi King Salman informed Russia his kingdom wants the Syrian president out of office
Watchdog slams UN for appointing Saudi Arabia to women rights body

Cluster: 66aq06
US intercepts 2 Russian bombers off Alaska's coast
Nato intercepting highest number of Russian military planes since the Cold War as 780 incidents recorded in 2016

Cluster: 667iuua
Trump issues trade warning to Canada: "It's another typical one-sided deal against the United States and it's not going to be happening for long."
Angela Merkel reportedly had to explain the 'fundamentals' of EU trade to Trump 11 times

Cluster: 6686v4
Trump administration certifies Iran is complying with the nuclear deal Trump once called the 'worst deal' ever negotiated
Ukraine president asks US to maintain sanctions against Russia.

Cluster: 66vih
A jailed critic of Philippine Duterte praised police who alleged in a Reuters report that officers received cash for executing drug suspects, and said an international criminal case should be filed ag
Champs Elysees in Paris closed, reports of 2 police officers shot.

Cluster: 6798b
Four More Secret Jails Illegally Holding Gay Men Discovered in Chechnya
Hayley: Anti-gay abuses in Chechnya 'cannot be ignored'

Cluster: 66e06
Venezuela crisis: Teenager and woman shot dead at anti-government protests
Venezuelan president Nicolas Maduro donates $500,000 to Trump fund despite economic woes

Cluster: 66gdaw
Palestinian sisters crossing into Israel for cancer treatment caught smuggling explosives
Lebanese leader calls for permanent ceasefire with Israel
8.2.6 driver.sh
A bash wrapper script calls each Python script sequentially every 12 hours.

8.2.7 populateTable.py
This is the Python script we used to generate a list of over 50,000 tagged entities from the Stanford named entity recognizer (SNER). In order to generate the list of tagged entities we first make a connection with the database to be able to query the content from every Reddit article that is stored. Next we use a SQL statement to get the content from each Reddit article. The content is then passed to the named entity recognizer which creates a list of tagged entities. The following screenshot shows the raw data that SNER produces.

```python
```

Figure 8.5: Results from SNER
This data set contains every word from the Reddit article with a tag of ‘O’ (other), ‘PERSON’, ‘LOCATION’ or ‘ORGANIZATION’. We then clean the data by removing entities with the ‘O’ tag, group together entities based on locality, and count the frequency for each entity. The following screenshot shows the cleaned data after grouping related entities, removing unnecessary entities and counting frequencies.
With the cleaned data, we then proceed to populate the tagged entities table in the database with a simple SQL statement. Each entity has a name and a tag, the name is the entity’s name and the tag is the type of entity it is (person, location, etc.). Figure 8.7 shows the tagged entities table for the entity ‘Facebook’.

<table>
<thead>
<tr>
<th>name</th>
<th>tag</th>
<th>frequency</th>
<th>month</th>
<th>year</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook</td>
<td>LOCATION</td>
<td>1</td>
<td>3</td>
<td>2018</td>
<td>2018-03-24 19:26:14</td>
</tr>
<tr>
<td>Facebook</td>
<td>LOCATION</td>
<td>1</td>
<td>4</td>
<td>2018</td>
<td>2018-04-03 11:23:52</td>
</tr>
<tr>
<td>Facebook</td>
<td>LOCATION</td>
<td>1</td>
<td>11</td>
<td>2017</td>
<td>2017-11-05 13:13:33</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>120</td>
<td>1</td>
<td>2018</td>
<td>2018-01-01 04:44:42</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>109</td>
<td>2</td>
<td>2018</td>
<td>2018-02-03 14:24:52</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>523</td>
<td>3</td>
<td>2018</td>
<td>2018-03-05 05:21:09</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>42</td>
<td>4</td>
<td>2017</td>
<td>2017-04-04 04:57:16</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>124</td>
<td>4</td>
<td>2018</td>
<td>2018-04-01 00:22:13</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>74</td>
<td>5</td>
<td>2017</td>
<td>2017-05-01 02:42:16</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>57</td>
<td>6</td>
<td>2017</td>
<td>2017-06-01 16:13:38</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>25</td>
<td>7</td>
<td>2017</td>
<td>2017-07-04 11:51:20</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>31</td>
<td>8</td>
<td>2017</td>
<td>2017-08-01 21:19:40</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>105</td>
<td>9</td>
<td>2017</td>
<td>2017-09-05 12:12:28</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>94</td>
<td>10</td>
<td>2017</td>
<td>2017-10-02 08:06:03</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>175</td>
<td>11</td>
<td>2017</td>
<td>2017-11-01 05:00:54</td>
</tr>
<tr>
<td>Facebook</td>
<td>ORGANIZATION</td>
<td>59</td>
<td>12</td>
<td>2017</td>
<td>2017-12-01 04:24:34</td>
</tr>
</tbody>
</table>

Figure 8.7: Tagged entity database table shows the tagged entities with frequency value and date timestamp

8.2.8 google-trends.py

This is the Python script to generate a series of ten trend graphs from Google. First the tagged entities database is queried for the top ten trends of the week. The query returns a list of the most frequently referenced keywords from Reddit articles over the past
week. These ten keywords are then passed into the pytrends wrapper in order to generate a series of points. Google returns the trend as a series of data points with the popularity of the keyword mapped to the date of occurrence. We then take these mapped values and use a Python graphing library known as pygal to generate the trend graphs [14].

8.2.9 reddit-trends.py
This is the Python script used to generate a series of ten trend graphs from the WorldNews subreddit in Reddit. First the tagged entities database is queried for the top ten trends of the week. The query returns a list of the most frequently referenced keywords from Reddit articles over the past week. These ten keywords are then used in ten different SQL statements to retrieve the keyword frequency by month over the past year. Once the SQL statements are returned, we use the data to create ten distinct trend graphs of the data. The trend graphs are created using the Python graphing library pygal [14].

8.3 Front-end Trend Display code
The display portion of this project is split into two separate web pages. For the purposes of easier display on the two monitor setup that is being used. The information about the trend and article displaying code will be discussed here.

8.3.1 .htaccess
.baiduaccess manages redirects for the website. It also makes calls to actions defined in the PHP controller so JSON data can be retrieved and passed to HTML where it is needed [1]. This must be turned on in your Apache configuration to allow the site to operate properly.

8.3.2 config.php
This is the configuration file for the website. Inside, the system path for the website, the URL, and database access information is defined. This file is unviewable to visitors who inspect the website [1].

8.3.3 global.php
This defines config.php and auto-loads objects defined in the model [1].
8.3.4 siteController.php

The sole controller for the website, defining actions that need to be accessed in order to manipulate information in the database so it can be displayed in visualization [1].

- Home: This action is used to include the home.tpl template so it is displayed when someone loads the website.
- GetArticles: Retrieves all the articles in a given cluster and parses them to JSON data, to be used when populating the article carousel.

8.3.5 home.tpl

This is the template webpage file that the server displays when someone visits the homepage. This is the only viewing document required since the website is a single page application. The extension .tpl functions like an HTML document [1].

- Bootstrap: The website relies on the use of Bootstrap as a framework[17].
- Modal Views: Pop-ups that display information to users when clicking on elements.
- Inline PHP: Connects the front-end viewing document to the back-end PHP. Can be found displaying the actual cluster and article data.

8.3.6 public/

Directory containing publicly accessible files for the website. Files include images, javascript, and CSS documents [1].

8.4 Cluster Display Code

8.4.1 ball_animation_1.js

This file contains the driving script behind the animated cluster display. The circles are drawn after taking a title from the cluster database. This information is gathered using cluster.php and parsed using Ajax. It uses setInterval() to animate each cluster and is currently running at a delay of 16ms (about 60fps), this can be easily changed to run slower for less powerful machines. The code draws a circle and then moves it based on its randomly generated velocity. The velocity is currently a value between -1 and 1, so every 16ms the ball will move between -1 and 1 pixel in the x and y position. The balls also operate under the principle of perfectly elastic collisions with equal mass, so when two balls collide they will essentially swap velocities. Hence, Circle1.vx will now equal Circle2.vx. This code can easily be modified to give each circle mass (potentially based on the centrality of the cluster).
8.4.2 cluster.php
This file reads the entire cluster table in the database and creates a JSON object out of the data. The header of the file is also changed to the JSON type; this means that when the file is parsed using Ajax it can successfully interpret the file as a JSON rather than a PHP file.

8.4.3 index.php
This file is the default file that is read by PHP it simply contains a practically empty HTML file which contains the canvas object that will draw the bouncing circles, added by the ball_animation_1.js script.

9 Lessons Learned

9.1 Use Existing Tools
Throughout the project, we have learned several important lessons. One of the biggest lessons is to use existing tools. In this project we use many Python libraries, many of which have been created by experts in their fields. Using their vast knowledge and experience is very helpful and greatly reduces the time the project would have taken otherwise, while at the same time increasing the accuracy and speed of our work.

9.2 Start Early
Another very important lesson we learned was the importance of starting early. We have run into several unexpected issues using unfamiliar technologies and they slowed us down considerably. Had we started our work sooner we would have been in a much better place.

9.3 Research
We also learned the importance of research. For the most part none of the members of our group are particularly knowledgeable about the different types of statistical modelling, so when we needed to update the legacy code we were working on improving we barely understood anything. Only after lots of research were we able to fully comprehend what was being done.
9.4 Regularly Scheduled Meetings

This project has taught us the importance of regularly scheduled meetings. For the most part, all of the projects we have completed in college could have been completed in one or two sessions; this project, however, required a lot more time investment, that combined with our busy schedules meant that having regular meeting time was very important.

9.5 Documentation

The way this project played out taught us how important documentation is to understand code done by previous people. Starting this project was incredibly difficult due to lack of understanding of what settings the previous project group had configured for their python scripts. The time spent trying to setup the settings and the database took so much time that could have been cut down if the previous group would have documented their settings properly.
Acknowledgments

We would like to thank our client Liuqing Li for being very helpful throughout the project thus far. We would also like to thank Dr. Fox for the guidance and experience he has given us. Finally we would like to thank the previous GETAR groups. Without their work we wouldn’t be able have been able to get as far as we have.

**Liuqing Li:**
liuqing@vt.edu

The client for the Event Trend Detection project who helped provide us with input and feedback for each step we were required to take. Additionally, he was always available anytime for a team meeting where we would converse about the progress of the project and provide guidance when needed.

**Edward Fox:**
fox@vt.edu

Professor Edward Fox guided us through the separate phases of the project. He gave us distinct feedback and help when needed. He was always open to discuss our problems and concerns during every step of the way.
References

Appendices

Appendix A Milestones and Timeline

A.1 February

A.1.1 Milestone 1 (01/26 to 02/09):

- Overview: We plan on researching during this time period. The research process will include looking through previous project documents/code, and coming up with ideas and discussing them with the client for improvements to the project. We also plan to move the code base to Gitlab for better code management.
- Deliverables: Have the project code in a private repository in Gitlab. Additionally we will have improved documentation of the project which includes the creation of a README. We plan on sharing the private repository with the client.

A.1.2 Milestone 2 (02/09 to 02/23):

- Overview: We plan on finishing the research and starting the testing of the actual project within a local environment. During this period we plan on gathering a better sense for the project to begin development. Additionally we plan on starting the development of the trend detector.
- Deliverables: Have the project running within a local environment.

A.2 March

A.2.1 Milestone 3 (02/23 to 03/09):

- Overview: During these 2 weeks we plan on starting development on the front-end and clustering portion of the project. Additionally we will continue working on development of the trend detection portion. Lastly we will discuss the clustering implementation with the client.
- Deliverables: Have a finished design and completed decisions for the improvements we are going to make. Have another chosen source for the news (such as Google News) in addition to Reddit.

A.2.2 Milestone 4 (03/09 to 03/23):

- Overview: During these 2 weeks we plan on continuing development on the front-end and clustering portion of the project. Additionally we will continue working on development of the trend detection portion.
● Deliverables: Add the trend table to the database. Have an implementation plan for the user interface and clustering algorithm.

A.3 April

A.3.1 Milestone 5 (03/23 to 04/06):

● Overview: During these 2 weeks we plan on continuing development on the front-end and clustering portion of the project. Additionally we will continue working on development of the trend detection portion.

● Deliverables: Add data to the trend table in the database. Implement the new clustering algorithm. Improvements to the design of the user interface will have been made.

A.3.2 Milestone 6 (04/06 to 04/20):

● Overview: During these 2 weeks we plan on continuing development on the front-end and clustering portion of the project. Additionally we will continue working on development of the trend detection portion.

● Deliverables: Implement the scraping of the news sources to form a list of trends. The clustering algorithm will be improved. The design of the user interface will be improved.

A.3.3 Milestone 7 (04/20 to 05/02):

● Overview: During these 2 weeks we plan on finishing development on the front-end and clustering portion of the project. Additionally we will finish working on development of the trend detection portion.

● Deliverables: The changes to the user interface will be complete. The improvements to the clustering algorithms will be finished. The trend detection and trend table will be completed.

Date that client has signed off on this: Liuqing Li (Approved, Feb 2)

Date that instructor has signed off on this: 2/3/2018 @ 21:35
## Appendix B Completed Work

<table>
<thead>
<tr>
<th>Date</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/23</td>
<td>Meeting with Liuqing</td>
<td>Discussed about the project goals and possible milestones.</td>
</tr>
<tr>
<td>01/26</td>
<td>GitLab setup</td>
<td>Setup a repository in GitLab for version control.</td>
</tr>
<tr>
<td>01/30</td>
<td>Milestone composition</td>
<td>Created milestones for every two weeks.</td>
</tr>
<tr>
<td>02/04</td>
<td>GitLab setup complete</td>
<td>Added all necessary files to the repository.</td>
</tr>
<tr>
<td>02/10</td>
<td>Research phase started</td>
<td>Started looking over previous group’s code and brainstormed design.</td>
</tr>
<tr>
<td>02/14</td>
<td>Local machine setup</td>
<td>Began working with the project in a local environment by installing dependencies.</td>
</tr>
<tr>
<td>02/20</td>
<td>Trend table created</td>
<td>Created a trend table in the database on the local machine.</td>
</tr>
<tr>
<td>03/01</td>
<td>Finished local machine setup</td>
<td>Finished setup of project in a local environment.</td>
</tr>
<tr>
<td>04/15</td>
<td>Finished trend detection</td>
<td>Finished trend detection with google and reddit news sources.</td>
</tr>
<tr>
<td>04/16</td>
<td>Finished improvement of clustering</td>
<td>Finished improving the clustering algorithm.</td>
</tr>
</tbody>
</table>
## Appendix C Table of Routines

<table>
<thead>
<tr>
<th>Routine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reddit-trends.py</td>
<td>Gathers the top ten trends from Reddit for the week and creates ten graphs for the data over the entire year from a database of Reddit articles.</td>
</tr>
<tr>
<td>google-trends.py</td>
<td>Gathers the top ten trends from Reddit for the week and creates ten graphs for the data over the entire year from Google trends.</td>
</tr>
<tr>
<td>poller.py</td>
<td>poller.py is responsible for scraping the ten “hottest” links off the WorldNews subreddit and storing the information gathered into the raw database.</td>
</tr>
<tr>
<td>populateTable.py</td>
<td>populateTable.py populates the database with a series of tagged entities from the current Reddit article content.</td>
</tr>
<tr>
<td>processNews.py</td>
<td>processNews.py extracts information from the raw database table, word-tokenizes it, and stores it into a NewsArticle object.</td>
</tr>
</tbody>
</table>