TimeMiner

Mohammad Ahmadzadeh

Alok Hota

Tyler Plunkett

CS 494/594 – Fundamentals of Digital Archeology

Introduction

Original Idea

TimeMiner was originally conceived as a project that would analyze various aspects of Wikipedia articles and generate a timeline of notable events. Events would be defined by *clusters* of articles which were determined to be related to the event. We wanted to also generate an interactive visualization of the clusters in the timeline to show how they would change and evolve over time. Aspects of interest for articles were edits, views, locations, time, and content.

This was a very broadly defined project and was too large to take on during the month's time in which we had to complete it. We had to narrow the scope of the project in multiple ways.

New Idea

The first method of reducing scope was to ignore edits. We found that in order to analyze the edits for articles, we would have needed to download a Wikipedia snapshot that included history. This would have been a massive download and would have also been unreasonable to parse given our time and resources. Additionally we found that edit information was quite inconsistent, and would have been a difficult source of useful information.

We also had to reduce the time scale of the project to include only information on articles in 2014. That is, the statistics on page views and link information came only from 2014, but the article may have been created before then. Finally, rather than algorithmically generating central articles for the clusters, we cherry-picked three *seeds*, or central articles, for the clusters: *2014 Winter Olympics*, *Malaysia Airlines Flight 370*, and *Ebola virus disease*, to try and analyze the respective events.

Data Retrieval

Page Links

In order to analyze the links in a cluster, we needed to find articles that linked into the seed and articles that were linked from the seed. These are *ingoing* and *outgoing* links, respectively. To find this, we used two SQL page dumps provided by Wikipedia: *pagelinks* and *pages*. The *pagelinks* table included source and destination articles for each link. However the sources used article IDs and the destinations used article titles. In order to translate between the two, we needed the *pages* table, which mapped IDs to titles. These files were rather large; *pagelinks* was 30 GB and *pages* was 4 GB.

Page Views

Wikipedia also provided statistics on page views via large compressed files. These files contained every article, media file, or WikiProject page that was accessed within an hour. For example, *pagecounts-20140101-000000.gz* contains information on any page accessed between 00:00:00 and 00:59:59 on January 1st, 2014. Each line contained information for a page: the language of the page, the title, the number of times it was accessed that hour, and the total data transferred. These files were extremely large. For 2014 (up to November 13th, 08:00:00), the files totaled to 946 GB when compressed.

Page Contents

We found a Python module written by Jonathan Goldsmith to access Wikipedia articles. It is a wrapper for Wikipedia's API. We used this module to access content, references, and some link information for articles. Wikipedia requests users to not crawl their site, so using this module helped access information quickly via their API instead of mass HTTP requests. However the module sometimes contained incorrect article titles and link information. Additionally, the content had some Markdown syntax and section headers and other extra data.

Data Analysis

Link Analysis

We tried importing the tables from the SQL dumps and various optimizations to speed up the process. However they ran for nearly a week without completing. We then decided to use regular expressions to parse the dumps and extract the information without creating a MySQL table. This was much faster and completed in approximately 20 minutes.

From here, we were able to find ingoing and outgoing links for each seed. We kept each list of links in a set and took the intersection of them to find *backlinks*. This means that the seed contained a link to an article, and that article had a link back to the seed. We originally thought this would result in manageable clusters, but they actually resulted in clusters ranging from 64 – 285 articles for the three seeds.

View Analysis

The first step to analyzing page views was to reduce the amount of information in the page view statistics files. Since the files were already large when compressed (approximately 80 MB each), we used *zcat* to get the output of the file without uncompressing the file to disk. From here, we eliminated any entry that was not from the English language Wikipedia.

We could now get focused information on page views since we had lists of specific articles to analyze from the link analysis step. We generated year-long lists of page views for each article in each of the three clusters. These lists had 7,592 data points – one for each hour from January 1st 2014 00:00:00 to November 13th 2014 08:00:00. Each month took around 6 hours wall time to analyze, so we ran separate processes for each month to analyze in parallel.

Content Analysis

Since the clusters for each seed were still too large, we needed to *cull* them by analyzing article content and references. This would determine which articles were more related to the seed than others. We first analyzed articles with common references. We kept any article that had at least one reference source in common with the seed, and eliminated all others. This reduced the cluster size quite a bit (90% in one case), to a range of approximately 20 – 33 articles.

Additionally, we analyzed the content of each article compared to the seed. In order to remove bias against longer articles, we looked at the top ten words in articles instead of getting word counts. For each article in the cluster, if that article's top ten list contained at least 3 words in common with the seed's top ten list, we kept it. We eliminated articles with 2 or fewer words in common. This reduced the cluster size range to 7 – 15, which was our within our ideal range.

We also used these top ten lists to pick *event words* for each cluster. The top four most common words across all articles were chosen for this. We did this because we felt these words would reflect the general concept of the event across all of the closely-related articles.

Results

To solidify a concept in this project: we originally wanted to have a program that would algorithmically choose seed articles. That is, it would choose articles it felt were likely candidates for being central sources of information for an event. From there, it would generate a cluster of articles surrounding that seed. Because this was too large of a task for the project, we chose seeds and then wrote the cluster generation portion. However we want to analyze the results of the cluster generation as if the seeds were picked by the script. Additionally, since we chose the seeds, we can then analyze whether or not our generation is correct and follows what we know from this past year.

The three seeds we chose were for three different types of events. The 2014 Winter Olympics was a month-long event, the Flight 370 disappearance was a developing news story of an incident, and the Ebola disease outbreak was a viral (literally) news story.

2014 Winter Olympics

With the 2014 Winter Olympics, the script generated the event words *Olympics*, *2014*, *winter*, and *Olympic* (in order from most to least common). What is interesting about this is that “2014 Winter Olympics” is both the name of the seed article, but is also the name of the entire event. Thus it makes sense that the event words provide little more context. Two other words of note from the top ten words in the clustered articles were Russian and Sochi, which provide some locational context.

Interestingly, however, we found articles included in the cluster that were certainly collateral to the main event. The Olympics were held in February 2014 (as the visualization clearly shows), but a small bump in readership occurred later in October as a Formula 1 race took place in the same location as the Olympics. We had hoped to find collateral events like this from the beginning, and were glad to see this result.

Malaysia Airlines Flight 370 Disappearance

Conversely, the Malaysia Airlines Flight 370 seed generated the event words *flight*, *Malaysian*, *airlines*, and *Malaysia*. Though these event words are similar in nature to the previous seed, they are less useful in defining the event. There was surprisingly not much mention of disappearance in the top ten words. The word “search” appeared twice in the top ten, which would have hinted at something being missing, but there would still be a lot of context missing.

However, we can find more information about the event by looking at the cluster. For example, one of the articles closely tied to the seed was *Malaysia Airlines Flight 370 unofficial disappearance theories*. This clearly suggests that the flight went missing, and hints at the possibility of it still being lost (which it is).

Ebola Epidemic

We found this to be our best result. The seed, *Ebola virus disease*, tells us that something related to the disease has happened, and we can infer anything from there, but the event words clearly summarize the event: *virus*, *Ebola*, *disease*, *outbreak*. Further details of the event can be found in the cluster itself, with article titles like *Ebola virus epidemic in West Africa* and *2014 Ebola virus epidemic in Guinea*, which give locational context again.

What is especially interesting is that none of the articles directly mention through titles that the Ebola epidemic crossed the Atlantic Ocean to the US or crossed into mainland Europe. However, when patients with the virus were confirmed to have it in the US and Europe, the cluster's page views spiked extremely high. This is interesting because despite it not being a widespread problem in the US, it became a highly viewed topic on Wikipedia, where most of the information pertains to African countries where the disease has been difficult to control.

Visualization

We utilized streamgraphs in our visualizations of the three clusters. Streamgraphs are a variant of stacked area graphs and were invented by Lee Byron in 2008. We used processing.js, a Javascript port of the Java-based Processing library, for the generation of the graphs. This allowed us to have an interactive visualization that lets the user hover over elements in the stream to get information on the articles in the cluster. The visualizations can be viewed at the following URLs:

- http://web.eecs.utk.edu/~mahmadza/streamgraph.js/olympics.html
- http://web.eecs.utk.edu/~mahmadza/streamgraph.js/mh370.html
- http://web.eecs.utk.edu/~mahmadza/streamgraph.js/ebola.html

Hovering over elements in the streams will show the article, time step, and value for that article (i.e. page views) in the top left corner. At the bottom of the page are various options to change the visual look of the graph. Left-clicking and dragging horizontally over a portion of the graph will zoom in on the selected area, and pressing Escape resets the zoom level to default.

There are a few improvements that need to be made to the visualization. First, the colors currently do not signify any importance or attribute of the articles in a cluster; they are randomly chosen. When zooming in, the current time step indicator does not update to reflect the new view.

Conclusion

We are overall happy with this first draft of the project, but have identified areas that need to be improved upon. The clusters generated with the three given seeds contained interesting and unexpected articles, from which we can infer many things. For example, we can see from the cluster articles for the 2014 Winter Olympics that there is a relatively low spread of topics. We find articles on certain countries' participation in the Olympics, past Olympic events, and the location in which the events were held (which gave us the collateral result caused by the Formula 1 race).

With Flight 370, we see a slightly large spread of article topics, ranging from the flight itself to the type of plane, to conspiracy theories. What we find interesting about these articles included in the cluster is that they are all fairly inconclusive with details regarding the story of the flight, which was reflected in the media's abrupt handling of the story.

We would like to refine the method by which we cull the clusters during processing to include more complex content comparison and reference comparison. Wikipedia's articles should be considered as a highly connected graph. What we have done is include only articles that are one link away from the seed in our cluster, or subgraph. It would be interesting to generate a metric that takes into consideration articles at a greater distance from the seed.

Finally, we would like to revisit and attempt the original idea of parsing the entirety of Wikipedia's articles to predict which articles would be seeds. Given certain parameters that we define as central to an event, we can see what events may have been overlooked or overemphasized in mainstream media that still exist in written record.