CS1951a: Final Project

Summary

News media has always been a race for first to publication. In the past, publishers had hours or days to track down stories and their sources, check facts, produce written content and go print. In modern news media, that window for success has shrunk to minutes if not seconds. More so than ever, readers have little loyalty to their source, moving wherever the story first emerges. If the Washington Post gets a push notification out to mobile even a minute before the New York Times, the Times has lost the first mover's advantage; a serious blow to viewership. As the New York Times editor for newsroom strategy says, "Anytime we push our news directly to a user's mobile, we see clicks and swipes skyrocket". And in breaking news you come in either first or last.

This rapid-paced publishing environment presents several very serious challenges. The first and most crucial hurdle is story discovery. Newsrooms sit in the dark, waiting for a lead to come their way. They employ manpower to find stories, scouring Tweets, Facebook posts, police scanners and other information vectors by hand. The publisher that gets lucky gets the scoop. With readers pawing at their lockscreens for whichever push notification pops up first, seconds can make a difference.

With a story discovered, next comes fact checking. The pressure to write and publish as fast as journalists can source and synthesize often pushes the focus on fact checking to the second tier. The fallback is to publish as little as necessary at first, to get first mover's advantage, and then supplement with more robust material when facts come to light. This leaves a lot to be desired in the context of the typical high quality reporting of publishers like the Times or the Post.

Finally, with a story sourced and written, a value must be placed on the story. Does it get pushed to the homepage of the website? Of the app? Does it warrant a push notification? These seemingly small decisions can make all the difference in the success of a story. Bother a user with a useless notification, and you cause annoyance, and in the worst case, the loss of a user. At the very least, the sorting of stories is an indicator to readers of the priorities of the publisher.

The advantages of automating this complex process promises not only faster response time for publishers, but also higher quality, better curated material for readers. In the interest of automatization, we propose two hypotheses aimed at identifying emerging stories and predicting their importance to readers.

Hypotheses

In densely populated urban areas, there exists a statistically significant correlation between a sudden increase in volume of Tweets sharing similar timestamps, geolocation, and content (homogenous Tweets), and a single event with importance to a broader public.

The magnitude of increase for homogenous Tweets is a statistically significant predictor of the popularity of news articles written about the event that catalyzed the increase.

Experimental Procedure

Story Identification and Synthesis

1. Given a corpus of geotagged Tweets from a densely populated urban area, parameterize the Tweets based on the following features:
    a. timestamp
    b. geolocation
    c. content
2. Cluster Tweets using a clustering algorithm (such as k-means clustering)
3. Validate the clustering by matching the predicted clusters against published news articles in common sources such as the New York Times.
4. With a clustering technique identified and verified, modify the algorithm to cluster in real time, with n-dimensional samples (representing Tweets) placed into the model as they are sourced.
5. Using the best clusters identified from initial clustering, identify a series of benchmarks that are used to classify a cluster as a “story” or as an “anomaly”. An example of a story might be the recent gas explosion in NYC, while an anomaly would be Tweets that are uncorrelated in terms of content.
6. Using the corpus of historical Tweets, simulate the stream again, by feeding the Tweets to our clustering algorithm, delayed by the difference between the timestamps of Tweets. This allows us to see how our algorithm would have performed had it been running when we originally collected the corpus.
7. Evaluate the effectiveness of our algorithm by matching the timestamp at which our algorithm successfully classifies a Tweet cluster as a “story” against the timestamp of the first published article about that story. The larger the gap, the better the performance of our algorithm.
8. Use words and phrases common amongst the Tweets that make up the story to automatically produce a rough human readable synthesis of the story.

Story Importance Prediction

1. Given the same corpus as used above, and clusters of Tweets classified as a “story”, we will identify the published articles that correspond to the Tweets.

2. We will then seek to fit an n-dimensional function to the growth of the cluster of these Tweets.
3. With a function fit to the cluster growth, we will search for a mapping between the characteristics of the function and the popularity of the article.
    a. The goal is to identify characteristics of the Tweet cluster growth that enable us to predict the popularity of a story.

Data Source

To test these hypotheses, we need to collect Tweets from a geographic area over a long period of time. We are going to focus on New York City, and use the location Filter functionality of the Twitter API to sample based on a geographic bounding box surrounding NYC. To understand how much data is realistic to capture and analyze, we need to get a sense of the volume of Tweets coming from NYC. From that volume, we will be able to work backwards based on the amount of computing power we have on hand to understand over what period of time we would like to collect. Ideally, we will be able to obtain data over multiple 24 hour periods, with the goal of collecting Tweets that pertain to some sudden event within the city that would be both Tweeted about and be pertinent to publishers. With data collected and an event identified, we begin our process of finding the needle in the haystack, described above.

We will need to have our data collected by April 17th.

Data Cleaning and Integration

The Twitter data will be fairly organized, so not much entity resolution is necessary. What is necessary is analysis of the content of the Tweets so that Tweets relating to the same subject can be linked together.

As a preliminary approach, we imagine an initial clustering based solely on geographic area, perhaps a radius of 3 or 4 city blocks. All Tweets that occur within an area within a relatively small time period will be considered nodes of a graph. Nodes will be connected if the content of the two nodes is judged to be about a similar topic. These edges will be weighted based on 1/(time difference between Tweets) so that Tweets that were posted in a small time range are linked with a higher value.

Matching Tweets based on similar content will be very difficult. A first step will be looking for direct word matches, as well as perhaps a measure of emotion by analyzing punctuation or looking for other emotionally charged words. Beyond these preliminary comparisons we can use public datasets of synonyms or simply words that have similar meanings. We may turn to Mechanical Turk to create these datasets if they cannot be found publicly.

When a separate graph of Tweets grows to be a certain size, it will be classified as a story. These graphs will be visualized on a map of NYC as a transparent circle that grows with the size and weights of the graph. Given enough computing power, this would be done in real time so that stories emerging across the city can be visualized on one screen. Selecting a story indicator will bring up the Tweets contained within the associated story graph. A prediction regarding the popularity of an article written about the story, based on the characteristics of the Tweet graph, will be shown as well.

Deliverables

Stage 1

- Data collection
- Ability to “replay” collected data to simulate real world rates.
- Clustering of Tweets based only on geographic proximity and time.

Stage 2

- Prediction of anomaly vs. story based on Tweet proximity only.
- Clustering of Tweets based geographic proximity, timing, and content.
- Improved prediction of anomaly vs. story using content association, as compared to geographic proximity only.

Stage 3

- Prediction of story’s success based on characteristics of Tweet cluster growth.

Backup

It is very possible that we will not be able to collect enough data to make this project work, or the data we do collect will not absorb an event worthy of classification as a “story”. In that case, we will simulate a story by creating Tweets artificially and injecting them into the dataset to see if our algorithm will detect them.

It is also possible that we will not be able to identify Tweets as having similar meaning/content. In this case, we can resort to grouping Tweets based purely on geographic tags.

Furthermore, even if we are not able to find characteristics of Tweet clusters that map to the popularity of stories, we can present our approach to the problem and suggest future work that might achieve success.