Interesting Updates: An infrastructure for personal news update delivery

Authors: Ladd Jones and Mu-Fan (Leo) Weng

Motivation

It is hard to stay up to date on the topics we find interesting. While online users do have discernibly favorite sites that they visit, most of the time spent on those sites results in only a few articles being chosen to actually read. Although news websites each target a specific audience, the specific interests of those audience members vary. The motivation behind Interesting Updates is to cut out the time spent searching for articles a user is interested in and present them with those articles up front. The second piece of Interesting Updates is that the software will read the articles to you and also be able to summarize the contents of the article so that time is saved, and other tasks can be accomplished while listening. Scrolling around a website takes the full capacity of our hands while an audio source allows for other productivities with the hands.

Related Works

News360

- One well-known company that is already attempting to tackle the issue of personalized news aggregation is News360. This product asks users to select topics that they are interested in from a list of varying specificities and then presents the user with a list of articles they might find interesting. The News360 software analyzes users' interactions with articles on their website and then creates an interest Graph that determines which articles to present in the futures.

Freedly

- Freedly is another company in the market space that relies on RSS feeds (rich site summary) and social media integration to give users a personalized news aggregate. Freedly is more of an organization tool and not self-recommending software. It is praised for its minimalist design and multiple layout options. The main takeaway from Freedly is that you can have all of your online information organized in one space.

Panda

- Panda totes itself as an aggregator of the aggregators. Panda pulls from various news sites such as Github, Behance, and Product Hunt to give technologist an all in one update source. Panda allows users to browse multiple websites at the same time and is great if your interests are in the tech world. Personalization is also a big part of Panda's pitch, but it comes at the time and expense of the

users own tweaking. Lastly, Panda has done a good job with extensive search within feeds for users trying to find specific details.

General News Aggregation sites

- There are many online news aggregation sites that focus on specific topics such as photography, gadgets, politics, etc., and some even offer personalization. In terms of personalization, the current sites that aggregate news ask the user to tell them what they like and don't like, and in order to change these setting a user must manually adjust them.

Features and Differentiation

Interesting Updates customizes feeds to a personal level not seen by current companies, all without interfering with the user's time. Interesting Updates is an application that watches a user's online behavior and then presents them with information that is relevant without excessive searching. Unlike the previously mentioned companies, the user's time is not taken to customize the feed or update preferences. Interesting Updates will have the ability to pick up on a user's changes in interest by monitoring trends in searches and will then alter the information being presented without any input from the user. The input into the system is the normal browsing activity of a user.

The user receives this interesting and relevant information by asking the system for one of the multiple options. The first option will be the ability to have current related articles read to the user by simply typing in "tell me something interesting". Users can add articles to their favorites list to read later or skip the article altogether. There will be the article's text appearing on the screen but the goal is to free up the user's hand so they can complete other tasks while being updated. The second option that the user can ask the system will be to have a summary of events for different time periods such as World Events, Politics, Technology, etc., that are once again tailored to preferences set by the users past search history and other factors such as time of day.

At any point, while a user is listening to an article, the user has the ability to ask for a summary of the article and an NLP text-rank algorithm will give them a brief summary.

Data Aggregation:

- Users are not interfered with while data is collected
- Any online activity is taken into account, not just visiting a specific website but the articles themselves that are selected to be read by the user
- For each article, time of day, time spent reading, length of the article, and article metadata is collected

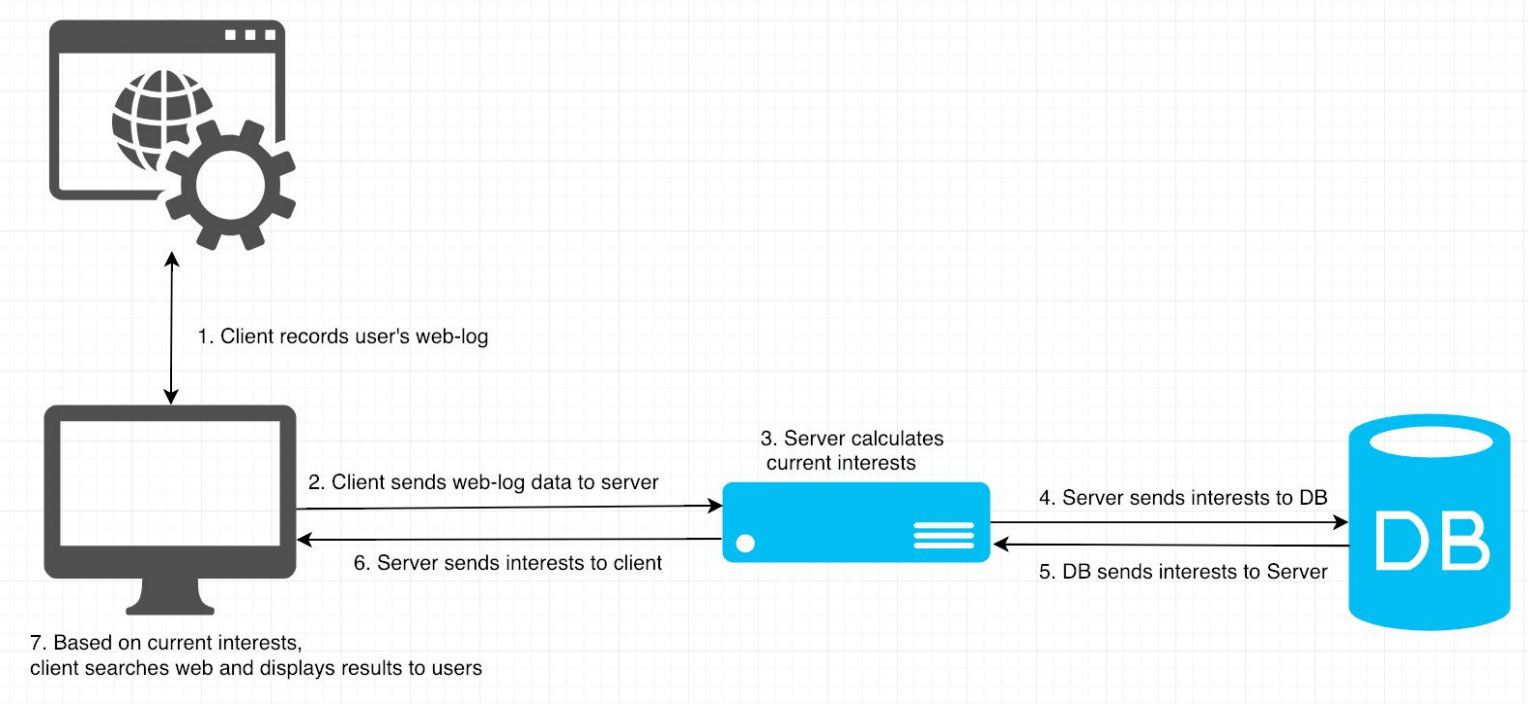
Data Presentation:

- Articles are presented via speech as well as being displayed on the terminal screen
- Articles can be summarized through NLP libraries
- A user can choose to save the article in text form to a folder on their computer if they want to save the article for later

Data Storage:

- User's online logs and any previously mentioned data is collected and sent to the server
- An article can be saved to a folder so that the reader can come back to it later
- A database stores the users interested and responds to queries with these interests

Future System Architecture



- Once web-logs are scraped from the user's computer the client sends the information to a server that then calculates interests. The specific interests of users depend not just on the articles they read but for how long, time of day read, and other personal information. Once interests are calculated they are stored in a database that is queried whenever the user requests information from the client application.
- Once the client application has the set of interests that it will search the web for it starts with the term most highly on the list and searches online for any related article.
    - Articles are chosen based on the same factors that are used to personally describe each user. For example, if a user usually checks a certain website for technical related inquiries, the client application will prioritize articles from this website. Articles within websites are once again chosen based off of interests of the user.
- Components
    - Backend: Our servers will be on EC2 through Amazon's AWS.
    - Database: SQL

    - Front end: Users will use the command line to interface with the software.
- Web-scraping data will be gathered from anonymous online archives.
    - To know what data to gather and where to look we are consulting this research paper https://pdfs.semanticscholar.org/14ab/cd056b8e8c41e551e0dc35f77504e65fed63.pdf
    - Our web-logs (datasets) will be gathered from the following websites:
        - http://archive.ics.uci.edu/ml/datasets/Anonymous+Microsoft+Web+Data
        - http://www.herongyang.com/Windows/Web-Log-File-IIS-Apache-Sample.html
        - https://www.researchgate.net/post/Where_can_we_get_free_web_log_server_data_for_performing_mining
- This software tool will accessible through the command line.

Datasets / References:

- For our data analysis, we would like to use the following dataset (link below). In summary, the dataset contains a list of user sessions online. Specifically, the sessions contain user ids, queries, query terms, URLs, their domains, URL rankings, and clicks. The Logs represent one month of activity and are sampled from a large city. There are 167,413,039 total records in the logs and 5,736,333 unique users. With this data, we plan to accomplish two main tasks in our demo. First is to be able to take the user's activity and create a model based off of the websites visited, number of clicks, and time of day. And second, build a scraping tool to survey the web for current articles that align with the user's interests and then present those articles to the user. The model will have to react changes in users preferences while the scraping tool must react to changes in the current news at all times.
    - https://www.kaggle.com/c/yandex-personalized-web-search-challenge/data
- What specifically does this dataset contain (this section below is from the data's user guide) https://academy.yandex.ru/events/data_analysis/relpred2011/#tasks
    - *SessionID* is a unique identifier for a user session.
    - *TimePassed* - the time elapsed since the beginning of the current session in conventional time units. The number of milliseconds in one-time unit is not disclosed.
    - *TypeOfAction* is a type of user action. This can be either a query (Q) or a click (C).
    - *QueryID* - unique identifier of the query.

- *RegionID* - unique identifier of the country from which the request is specified. We have included this identifier, since document rankings and relevancy scores may depend on the user's country (for example, for the query “Ministry of Foreign Affairs”). This identifier can take four values.
- *URLID* - the unique identifier of the document.
- *ListOfURLs* - a list of documents, ranged from left to right as they were shown to users on the Yandex issuing page (top to bottom).

  Example:

  10989856 0 Q 10364965 2 671723 21839763 3840421 180513 45660210 514963 41484044 3153206 1439919 4991367
  10989856 103 C 21839763
  10989856 955 Q 1009161 2 197515 197539 11 179526 5859272 1624306 1587784 1624296 5859294 2186374
  10989856 960 C 197515

  Each line in the file containing the estimates has the following format:

- The unique queries of each user will be analyzed to determine which sites are most important to the user and within those sites the type of article that the user is most interested in reading. Another piece of the project will rest on being able to classify websites into categories based on content.
- A complete guide for understanding the data-set we are using can be accessed here (must be translated from Russian).
  - https://academy.yandex.ru/events/data_analysis/relpred2011/#tasks
- The following website will allow us to understand weblog IIS and Apache files to allow us to parse these files to gather the specific information we need for the project.
  - http://www.herongyang.com/Windows/Web-Log-File-IIS-Apache-Sample.html
- Finally, these tools such as Chrome history view and export history will help us in gathering our own datasets by exporting our personal weblogs into TXT or CSV files.
  - https://www.nirsoft.net/utils/chrome_history_view.html
  - https://chrome.google.com/webstore/detail/export-history/hcohnnbbiggngobheobhdipbgmcbelhh?hl=en-US

Demo

- Our demo will focus on two tasks. Task one: process the dataset and generate a model of a hypothetical user's interest. Task two: demonstrate the adaptation of our software to current online news at a suitable refresh rate. This will be done by selecting a prediction model, meaning a machine learning to weight a user's online activity. Along with this, we used a natural language processing algorithm to map connections between articles so that the user's interests could be most accurately met.

Foreseen Risks

- Overly ambitious deadlines will not be able to be met on time.
- It will be difficult to calculate the true correlations between web searches and what the user is genuinely interested in.
- Security issues surrounding sharing personal information. People may not feel comfortable sharing all of their web histories. Some form of filtering needs to be implemented to not include personal information.
- Articles heavily based on multimedia (images and videos) may have to be filtered out or provide users with the link to access.

Deliverables

- Source Code for Project
- Presentation / Demonstration Materials
- Final Report

The work will be distributed equally between the two members, with Leo focusing on the server and Ladd focusing on the client web-scraping.

Item	Team Member
Client (web-scrapping)	Ladd

Server (interest calculation)	Leo
Database (storage)	Ladd, Leo
NLP (article summary and search)	Ladd, Leo

Work Plan

Week	Task
February 4	Project Proposal
February 11	Research web-log scraping
February 18	Be able to interface with data-set in order to process interests
February 25	Build the interest model from data-set
March 4	Set up server and backend
March 11	Develop the NLP algorithm
March 18	Set up web-scraping
March 25	Connect client and server/database for processing of user's online activity
April 1	Flex week
April 8	Presentations
April 15	Presentations
April 22	Deliverable April 20th

Future Reach goals

- Adding a web interface for easier access for users.

Implementation

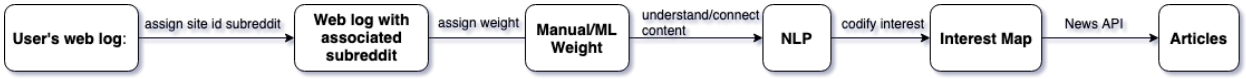


Fig. The pipeline of our implementation of the framework

Data Mapping

The approach that we chose for mapping the user's unique sources was to let a specific source map to a subreddit. The reason that we chose a subreddit mapping was that we wanted to have as much metadata associated with the user's search history as possible. Along with this, we wanted a consistent format for scraping a webpage so that we could implement a demo of our project without having to deal with websites handling metadata differently.

We assigned the most popular subreddits to the most popular sources, meaning that whichever websites the user visited most would be linked to subreddits with lots of users. The popularity of subreddits was judged based on subscribers. The websites that a user visits most often is going to be the one that contains most of their interests. This is not to say that we forgot about less frequently visited websites, it is that we wanted to have a subreddit that we new had frequent activity for the sites that would be most likely to end up being weighted highly by our machine learning algorithm.

The mapping ended up taking the first 1 million unique searches by the user and assigning them a subreddit. There are around 1 million + subreddits which is why even though there were 72 million + unique websites visited we stopped at 1 million. We stopped before using all of the available subreddits because as the bottom of the list is neared there are subreddits that are not and have not been active for quite some time and will not provide the type of metadata that we require for our interest mappings.

Weighting Approaches:

- Manual

In order to get a good idea of what interests the user likes exactly, we took into consideration a combination of the number of times the user clicked on this url (num_clicked), total time spent on that webpage (total_time), as well as the number of subscribers to that subreddit (num_subs). A simple linear combination algorithm of

num_clicked * (total_time / 100) * (log(num_subs)) allows us to predict the users' interest based on patterns visiting specific subreddits.

```
SELECT url, (num_clicked/1)*(total_time/100)(log(num_subs)) AS
weighted
            FROM (
                SELECT s.id, s.url AS url, COUNT(c.clicked_URL)
                  AS num_clicked,SUM(c.time_passed) AS
                  total_time, s.num_subs AS num_subs
                FROM (SELECT * FROM Clicks LIMIT 100000) AS c
                LEFT JOIN Subreddits AS s
                ON c.clicked_URL = s.id
                WHERE c.clicked_URL <= 1082444
                GROUP BY s.id
            )
            ORDER BY weighted DESC
            LIMIT 10;
```

Fig. SQL Code snippet to of the manual weighted algorithm to generate interest

- Machine Learning

Machine learning would help and more accurately predict the interests of the user. We implemented a random forest algorithm to predict the number of clicks on a webpage given the time spent on that website. This is useful in logs where in-page clicks are not recorded but time spent on site is. According to our experiments, we were able to achieve 74.5% accuracy in predicting and a mean absolute error of 0.74. If given more time, we would definitely want to develop a more robust ranking algorithm for the users' interests, based on similar ideas as Google's PageRank algorithm.

Natural Language Processing and News Generation

The natural language processing includes multiple phases. First of all, we took the top ten interests from the users' highest weighted categories. Within these ten interests, we take the top five posts as well as the descriptions of the subreddit and compile them into one text file. This compilation allows us to get an idea of what the subreddit is about and what the interest of the user is. Note that we do not include image posts, video posts, and other non-text posts from the subreddit. We then pre-process this text file by eliminating unnecessary punctuations and stopwords or redundant words. Through Porter Stemming and Lemmatizing, we can generate the top keywords or phrases of the compiled texts. We also used a CounterVectorizer to generate key singular words,

di-grams, and tri-grams. After consideration, we decided that di-grams most accurately reflect the ideas presented in the texts. Using these di-grams, we search through Google News using the News API to get two articles per interest.

Github Link
https://github.com/leo-weng/interesting-updates

References

- https://news360.com/home
- https://en.wikipedia.org/wiki/News360
- https://feedly.uservoice.com/knowledgebase/articles/180131-what-is-feedly
- https://feedly.com/i/welcome
- https://en.wikipedia.org/wiki/Feedly
- https://themeisle.com/blog/news-aggregator-websites-examples/
- https://usepanda.com/