A Web Interface for summarized news feed

Web Framework: Flask, a web framework for python

Database : MySQL

Description : A web interface is one of the implementation for our project “Personalized News Feed”. This web interface allows the user to select the news sources such The Guardian, BBC, CNN etc. per his/her preferences. Once the user chooses a website and clicks on appropriate button, corresponding news from the source will be presented to the user in the form newsfeed. The important thing about the news is that, it is summarized version of the original news article. The summarized news just presents the important part of the news which user needs to know about.

Before viewing the newsfeed, the user must create an account on the web interface with his/her credentials such as email, full name and the password of any choice. The user credentials are stored in a MySQL database. Upon successful creation of their account, user may log in to the web interface.

Once user has logged in, they can choose to click on the “News” which will take them to the newsfeed. Then they will be provided with the different news sources. Upon clicking on the news source, A python library called “Newspaper” scraps the corresponding news website for the news articles. Once the article is downloaded by the python script, it is then summarized with the help of natural language processing library “NLTK” and this summarized article is then presented to the user in the form of text. This whole process of web scraping and summarizing the article is done in the background with the help of python script.

We are currently working on having a uniform webpage design for all the webpages. We also plan to include images in the newsfeed. But we face a challenge of resizing the images, to properly fit the newsfeed. In addition, we are also working on making our newsfeed scrollable.

The potential roadblock that we face is reducing the response time of the website. Whenever a user clicks on a news source, the python newspaper library fetches the article and summarized the article using the NLTK library. This process takes quite a lot of time because each time python library must communicate to the news source website. To reduce the response time, we plan to prefetch the data even before the user requests it. We plan to do this by having a python script do the prefetching of the data and store it in a file and whenever a user requests for it, we just load the data on to the newsfeed.

Text to Speech Using Responsive Voice

This is one of the implementation of our summarized news feed. In this implementation user gets to hear the audio of the summarized news which he/she selects. This implementation is built over the web interface. The implementation makes use of the JavaScript API Responsive Voice to convert the text into the audio. The news is passed as an argument to function responsiveVoice.speak() and the output we get is a playable audio.

As stated previously this implementation is built on top of the web interface, we face the same roadblocks of scaling the newsfeed to more than just one news article. Once we have scaled the newsfeed then we just have to pass the content of each news article to the responsiveVoice.speak() method.

As of now the system has only one audio button which plays the news. The potential roadblock could be adding audio button for each news article. We are yet to figure out how to approach this problem.

Another problem that we are facing is the responsiveVoice.speak() method cannot process many special characters. In particular, parenthesis, returns and newline characters seem to be the most problematic Hence, we need to make sure that before passing news content to the method, we pre-process it to get rid of any special characters.

One potential problem that responsiveVoice presents is that it must chop up the audio files it sends back to the browser into chunks on google chrome. This makes audio sound choppy for long form speech. We may investigate to see if any other text to speech solutions could be used that work around this problem. There are also some other browser dependent quirks to responsiveVoice that we have not personally encountered so far.

We plan to add audio features such as stop, fast forward and controlling the sound level. However, we are not sure how to add these features at the time being. We would need to research more on the Responsive Voice API to figure it out.

Publishing summarized news using Twitter API

In order to fulfil the social media distribution side of our news summary application, we decided to go with Twitter to publish news. We looked into Facebook, but the API for publishing text to pages was quite cumbersome and seemed quite taxing to actually implement.

We decided to write this part of the project in python. Since the server side framework all runs in python, it would be easier to integrate this code with that of the flask framework. The Twitter API has support for many python wrappers. We decided to go with twython since it had the most documentation available and a decent sized user base. It also supports features that may be useful in the future including the ability to upload images.

A Twitter account for our app was specifically created in order to publish news stories. We also used the account to access the Twitter developer page which allowed us to find the API keys. We then set up a small python script that allows us to publish text to our Twitter page by passing text to the python script.

We then tried integrating the news summarization API with the python script. The summarization API often times returns stories longer than 140 characters. Therefore, each post had to be split up into chunks and sent out in order. There were also problems about where to split the chunks. Often times words would get split in half between posts leading to text that was hard to read. This was solved by only allowing posts to split after a space character was found.

Currently, we are trying to find a way to run the twitter script in the background of our server at a certain time interval. Once news is pulled, summarized, and saved, we want the twitter script to be called and each summarized story to be published to our page.

Roadblocks include having to deal with timing of the news updates and the updates of the twitter page. Sometimes there are instances in which the news API updates but actually returns no data, which may be problematic for the Twitter script.

Also the formatting for multiple subsequent tweets about the same topic does not follow the normal twitter convention. This makes stories more difficult to read. It may or may not be possible to reply to your own tweet like we would like.