TANR (Twitter Augmented News Reader) Executive Summary

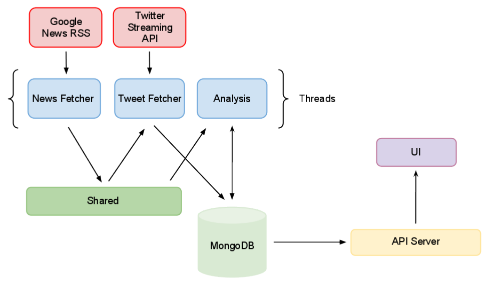
Rafał Szymański Maciek Albin Sam Wong Suhaib Sarmad Jamal Khan
{rs2909, mja108, sw2309, sss308, jzk09}@doc.ic.ac.uk

1 High Level Overview

Twitter Augmented News Reader - A webapp that pairs current news with relevant tweets.

Traditionally, the media has been presenting news stories biased with the writers' point of view or agenda, recruiting readers to join their agendas. This webapp presents a balanced view to the readers. We pair up relevant tweets to news emerging around the world. Our aim is to create a new kind of news reader that also presents the discussion behind the news.

2 Technical Overview



We decided to divide the project into two separate technical parts: *Analysis* and *UI*, that would be connected through our own *API* server

Analysis process is divided into three separate threads:

News Fetcher responsible for fetching current news stories. Getting information relevant to us and generating keywords

Tweet Fetcher responsible for fetching tweets using keywords generated by *News Fetcher*

Analysis uses tweets gathered by *Tweet Fetcher* and info gathered by *News Fetcher* to calculate Twitter's reaction to news stories

The API provides a very thin layer between the *UI* and the data stored in the database. It's written in Python using Flask and provides the calls returning results in JSON.

The UI connects to the server's API and displays the information in a tiled layout. These tiles dynamically rearrange themselves and adjust to the size of the users' window or screen. Each tile displays an image related to the article and title of the article. On hover/click of a tile, the image folds away revealing a short summary, a word cloud made from common words found in relevant tweets, a list of relevant tweets and the overall sentiment.

3 Software Engineering Issues

We opted to use Python as our primary language of development. This was dangerous considering the magnitude of the project however we learned a new language and related python technologies including Flask.

We also had issues with Twitter API limitations as we were only allowed a particular amount of API calls per hour to create user networks, this consequently caused us to steer the direction of the project to augmenting news with reaction from Twitter commenters.

We also learned a lot about working in teams and how teams can make or break whether a project is successful or not. We learned quickly that if we did not assign roles to particular people then a lot of time is wasted in people trying to cover all aspects of the project. To us it made more sense to assign particular parts of the project suited to the individuals and if problems were encountered at an individual level then the help of the group was at hand.

4 Validation and Conclusions

From time to time we asked for feedback on the user facing products. This was done so that we could make design and functionality choices. We talked to our supervisor Jeremy Bradley, and he liked our user interface but wanted us to add pictures for each news story.

We used Pylint4 to validate the Python code which we wrote. This was done to check for errors such as incorrect indentation, bad naming conventions for class names and functions, among others. Checking for errors and validating the UI side of the project was done using the W3 Validator.

We conclude that this project was a good exercise in learning software development techniques, and exposing problems related to working in a group.