https://doi.org/10.48009/2_iis_2014_350-358
Issues in Information Systems
Volume 15, Issue II, pp. 350-358, 2014

AUTOMATED PLATFORM FOR AGGREGATION AND TOPICAL SENTIMENT ANALYSIS OF NEWS ARTICLES, BLOGS, AND OTHER ONLINE PUBLICATIONS

Michael R. Grayson, Mercer University, michael.richard.grayson@live.mercer.edu
Myungjae Kwak, Middle Georgia State College, myungjae.kwak@mga.edu
Anthony Choi, Mercer University, choi_ta@mercer.edu

ABSTRACT

Identifying the prevailing sentiment at a certain point in time regarding a certain product, company, or topic can be a powerful asset in many contexts. It can offer an edge in financial analysis, for example, or identify how well a particular product is being received by a given demographic. For these reasons, much research is currently devoted to the topic of using computer analysis to identify sentiment in various media outlets through the Internet. The current paper describes an automated platform for such analysis, specifically targeted at analyzing news websites, blogs, and other online publications. The platform consists of several components which work together in an automated, high-level way to allow the user to obtain and analyze sentiment scores on a given topic within a certain date range. The researcher can employ the described toolset to download large volumes of research data from a wide variety of Internet sources quickly and easily without requiring a subscription to a news aggregation service.

Keywords: Sentiment Analysis, News Aggregation, Google Custom Search, Social Media, Financial Analysis

INTRODUCTION

In the age of Information Technology, knowledge is the coin with which progress must be paid. In order to take advantage of the vast and continuous flow of information, human beings must rely on Information Systems to filter and process data. Unfortunately, machines are notoriously inadequate at identifying qualitative properties of information such as intention, sentiment, and emotion: the very properties that humans value most and are most interested in identifying. As computation becomes cheaper and faster, methods must be developed to leverage machine intelligence in order to circumnavigate these issues and provide value to the user. This is particularly relevant in areas of endeavor that have been historically dominated by trend evaluation such as financial analysis, epidemic monitoring, political polling, and drug side-effect identification. This paper presents a toolset designed to address these issues by taking advantage of free and low cost internet services and computer resources to allow the researcher the ability to download large volumes of relevant textual data, store that data, and score it for sentiment concerning a given topic.

In this paper, we present our toolset, Data Aggregation and Topical Analysis of Sentiment (DATAS), for automatic data aggregation and sentiment analysis. This discussion includes a review of related research, a high level overview of the current system design, a primer on getting started with and using the Google Custom Search API, and information covering our current sentiment analysis methods and results. A simple sentiment analysis tool is presented which provides coarse sentiment evaluation using a general sentiment lexicon and rough scoring methods. In the final section we discuss future work, which will include adding more aggregation sources, enhancing the lexicon for sentiment analysis with topic specific vocabulary, moving linguistic analysis to a more advanced text processing platform, and correlating generated sentiment scores with real-world data for verification.

LITERATURE REVIEW

Sentiment analysis (or opinion mining) methods have been studied extensively since the decision-making of both people and business is deeply influenced by the thoughts of leaders and ordinary people. Feldman categorized sentiment analysis into five fields: document-level sentiment analysis, sentence-level sentiment analysis, aspect-based sentiment analysis, comparative sentiment analysis, and sentiment lexicon acquisition [4]. Especially, he acknowledged that sentiment lexicon acquisition is one of the most important areas of sentiment analysis and mentioned that the sentiment lexicon is the most crucial resource for most sentiment analysis algorithms [4]. He also introduced several applications for sentiment analysis such as the area of reviews of consumer products and services

and financial markets [4]. Liu identified the fundamental problem of sentiment analysis and introduced several sentiment and subjectivity classification methods and feature-based sentiment analysis techniques [12].

Pang and Lee also covered techniques and approaches that can enable opinion mining systems and address the new challenges raised by sentiment analysis applications [13] [14]. Especially, they focused on summarization of evaluative text and on broader issues regarding privacy, manipulation, and economic impact that can be raised by the introduction of sentiment analysis systems [13] [14].

As for applications of sentiment analysis systems, Antweiler and Frank studied the effect of messages posted on Yahoo! Finance and Raging Bull by using computational linguistics methods [1]. They used Wall Street Journal news stories as controls and found that stock messages help predict market volatility [1]. Their study results revealed that the effect of stock returns is statistically significant but economically small and also disagreement among the posted messages is associated with increased trading volume [1]. Das and Chen developed a method to extract small investor sentiment from stock message boards by combining different classifier algorithms based on a voting scheme [2]. Their research results showed that tech-sector posting are related to stock index levels and also to volumes and volatility. Devitt and Ahmad also showed that both the informational and affective aspects of news text affect the markets in profound ways, impacting on volumes of trades, stock prices, volatility and even future firm earnings [3]. Ferguson et al. focused on the topic-based sentiment analysis in the domain of financial blogs by using paragraph-level and document-level annotations and examined how additional information from paragraph-level annotations can be used to increase the accuracy of document-level sentiment classification [5].

As for sentiment lexicon acquisition [4] methods, Kanayama and Nasukawa proposed an unsupervised lexicon building method for detecting polar clauses, which convey either positive or negative tone [11]. Their system was constructed to acquire so called polar atoms, the minimum human-understandable syntactic structures that can indicate the polarity of clauses, based on the context coherency [11]. Their experimental results showed that the precision of the automatically acquired lexicon was 94% on average [11].

Godbole, Srinivasaiah, and Skiena presented a unique system for sentiment lexicon generation and sentiment evaluation of news and social media [6]. In their system, sentiment lexicon is generated algorithmically using path-based analysis of word relationships on WordNet. Sentiment scoring is adjusted based on word proximity. Their algorithmically generated lexicon showed a high degree of correlation with manually generated word lists [6].

SYSTEM DESIGN

The news aggregation and sentiment analysis system consists of four major components which work together in a high-level way to allow automated data aggregation and sentiment analysis, as can be seen in Figure 1. These

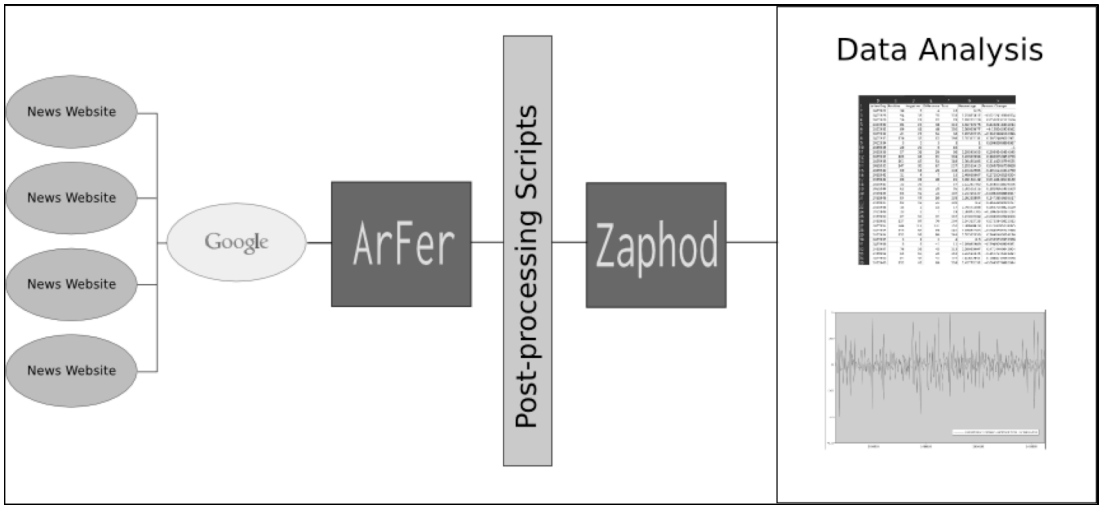

Figure 1: System Design

components include a customized search engine powered by Google Custom Search; ArFer (Java utility) to access this search engine for a given date range, parse its output, and download web pages that match the search criteria; a suite of Linux utilities to post-process and extract the text from these web pages and organize it for easy sentiment analysis; and Zaphod (sentiment analysis tool), to analyze and score the extracted text. The following sections describe these tools and how they function together to aggregate data and to identify and score topical sentiment.

CUSTOM SEARCH ENGINE

Using the Google Custom Search API custom search engine (CSE) creation tool, we created a CSE that includes 42 common and well known business blogs, news aggregation sites, and business analysis websites. Some examples of websites included are Forbes, Yahoo! Finance, and BusinessWeek. The following sections provide details about creating and using a CSE.

The Google Custom Search API provides a simple way to harness the power of Google search for a specific purpose. The API allows the user to specify a set of websites to be searched using Google's standard search algorithms. The search engine created by this process has a unique URL and can be accessed directly through the Google Custom Search website, or programmatically through JSON calls [7].

To create a CSE, you must access the Custom Search API home page:

http://www.google.com/cse

You will be asked to sign in with your Google account if you have not already done so. From this page you can create and edit your CSEs. Adding a CSE is a straightforward process. You must select which sites will be searched by your CSE and add them. In the past, this step had to be performed manually, but you may now create a CSE using keywords and categories pertaining to the subject you wish to analyze.

In order to access the CSE programmatically, you must first obtain the unique Search Engine ID created for you when you generated your CSE [7]. To find this value, access the "Edit search engine" option from the CSE home page. Look under the "Basics" tab for the "Details" option bar, shown in Figure 2. Select the "Search Engine ID" button and record the value that is displayed.

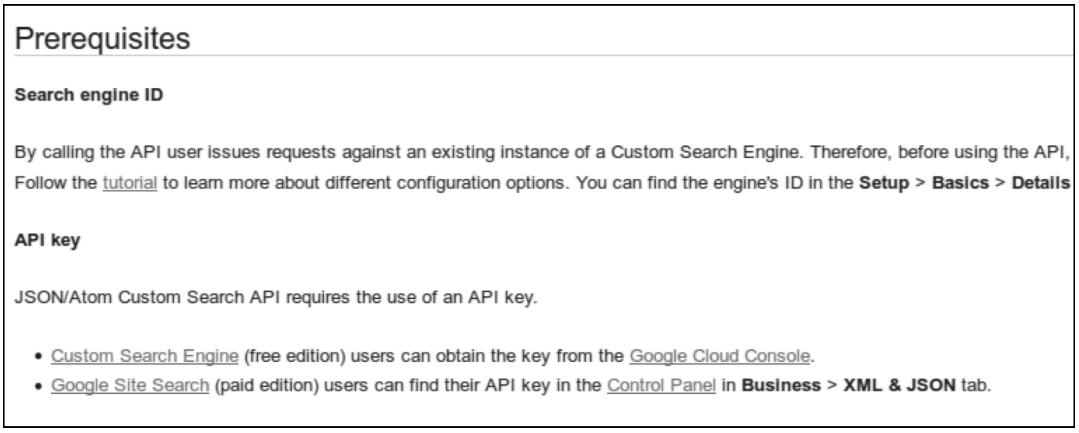

Figure 2: Search Engine ID and API Key

In addition to the Search Engine ID, you will need your Custom Search API Key to access your CSE [7]. If you have a paid Google Site Search account, this is done from the Google Site Search Control Panel, Business → XML & JSON tab. If you have a free CSE account, you will need to visit the Google Cloud Console, ensure that the Custom Search API is activated, and click on the APIs & auth → Credentials tab [7].

Once you have obtained your CSE ID and API Key, you can access your CSE programmatically using any tool capable of making an HTTP connection. The results returned over this connection will be in the JSON format and can be parsed using standard XML parsing libraries. In Java, this can be accomplished using the

HttpURLConnection/BufferedReader classes for accessing the CSE, and any of the various Java text and XML parsing facilities.

The CSE is limited to 100 free queries per day. A query is a single page of search results, limited to a maximum of ten URLs. A pay-for option is provided, which allows up to 10,000 queries per day, at a rate of $5.00/1,000 queries. The pay option can be enabled by adding a payment method to the account. The daily search limit can then be manually set from the default of 1,000.

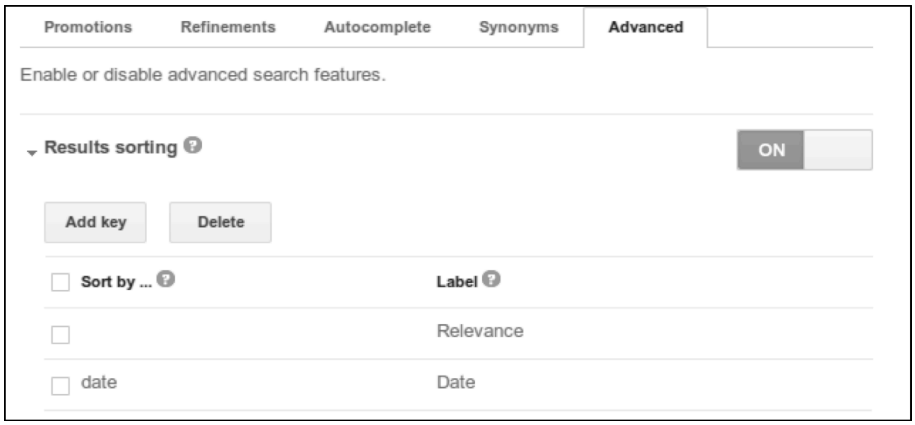

Figure 3: Sorting Results

The CSE is not initially configured to return results in ascending order by date. However, sort ordering by date is available as an option. This is important when performing queries on a specific date range, since the results are truncated at 100 results per search. Without the date ordering, articles from a particular date might be truncated without a clear indication that this is so, whereas with the date ordering, when a new date is encountered it is known that all the results from the previous date have been acquired. To set up sorting by date, one must access the Custom Search API console, under "Edit search engine" and click on the "Search features" option, shown in Figure 3. From there, select the Advanced tab, ensure "Results sorting" is set to On, and add a key with "date" in the key field, and an appropriate label [8].

NEWS AGGREGATION WITH ARFER

The Article Fetcher, or ArFer, is a Java utility for fetching news articles, blogs, and other textual matter matching a given topic. ArFer accesses the Google Custom Search API through JSON over HTTP. Although the Google Custom Search API does not directly provide a method for filtering by date, the JSON URL format accepts typical Google operators, including the "daterange:" operator. The "daterange:" operator accepts a date range in the Julian day format. ArFer converts a standard Java GregorianCalendar object to the corresponding Julian day, which is used to construct the complete URL. The results of the custom search are returned from Google in an XML format with brief summaries and result URLs pointing to the matching website. ArFer parses these results and, if available, downloads the full HTML of the website pointed to by the result URL. The HTML is stored in individual files for post-processing.

In order to access the CSE, a search URL must be constructed that contains the CSE ID and the API Key for your Google account. The custom search URL has the following general format:

https://www.googleapis.com/customsearch/v1?key=YOUR_API_KEY
&cx=YOUR_CSE_ID&q=SEARCH_QUERY+daterange:JULIANSTART-JULIANEND
+more:pagemap:article&sort=date:a&alt=json&start=STARTINDEX

The keywords appearing in all caps are placeholders defined as follows:

1. YOUR_API_KEY, YOUR_CSE_ID: The previously mentioned API Key and CSE IDs.
2. SEARCH_QUERY: A standard Google search query formed from a series of keywords joined by the "+" symbol.

3. JULIANSTART-JULIANEND: A hyphenated Julian date range. If the start and end dates are equal the search includes a single day.
4. STARTINDEX: A results index indicating where in the result set to start.

The "more:pagemap:article" operator instructs Google to try to restrict the returned results to websites that include metadata tags which indicate the content type to be "article" [9]. Although this method does not guarantee that the results will be articles, using it does net higher quality results. The "sort=date:a" operator tells Google to sort the results in ascending order by date, and "alt=json" indicates that the results are expected in JSON format.

In order to allow simple operation by the user, ArFer includes a facility for translating a user readable date (e.g. "January 1, 2014") into the Julian date index required by the Google Search daterange operator. Although the Java API does not directly provide a way to convert a date to the Julian date index, the algorithm for conversion is straightforward. The provided date string is parsed using the DateFormat.parse(String) method, which is then used to initialize a GregorianCalendar object. A pre-calculated epoch time corresponding to a known reference date is subtracted from the epoch time representation of the date to be converted, obtained using the GregorianCalendar.getTimeInMillis() method. The resulting difference is converted from milliseconds into days by dividing by 86,400,000 (24 hours/day x 60 minutes/hour x 60 seconds/minute x 1000 milliseconds/second). The corresponding difference in days is then added to the pre-calculated Julian date index of the known reference date.

The URL corresponding to a single result in the returned JSON data is given under the "link:" keyword. Parsing out this line provides a direct, unabbreviated URL of the given result. Keywords "formattedUrl" and "htmlFormattedUrl" are also provided, but are abbreviated and should not be used for accessing results.

Once the URL for a particular result is known, the raw HTML from the website is retrieved and printed to a file using the HttpURLConnection, BufferedReader, and PrintWriter classes. These are used to establish a connection to the URL, buffer the InputStream of the URL connection, and to print the data to a file, respectively.

POST PROCESSING SCRIPTS

A combination of AWK and shell scripts are used to reduce the HTML files downloaded by ArFer to a snippet of HTML containing only the news article of interest. The resulting set of simple HTML files are then converted directly to text using Aaron Swartz' html2text Python script [15].

In order to extract the text of the article body, the section of HTML code containing the article text must be isolated from the extraneous HTML of the website containing the article. Although there is not a perfect way of doing so, most websites use tags to demarcate related blocks of text in HTML. These tags often contain common descriptions of the content that can be parsed to identify a section of HTML code that corresponds to an article. An AWK script was created that can isolate a block contained within tags that has one of several descriptors such as "article," "article_body," "articleBody," "articleBodyContent," and so on. The output of this AWK script is a reduced block of HTML code that should contain the text of the article to be analyzed.

Figure 4: Zaphod Results and Scores

Once the downloaded HTML files have been reduced to contain a smaller amount of HTML code and the article body, the resulting HTML is passed through a Python script which strips the HTML and prints only the raw text. Since the files have been preprocessed to essentially contain only the article, the resulting text file is the article

body. Sometimes there is a small amount of extraneous text included with the article that consists of things such as stock quotes, but these non-sentence inclusions are easily identified and discarded in later steps of the analysis.

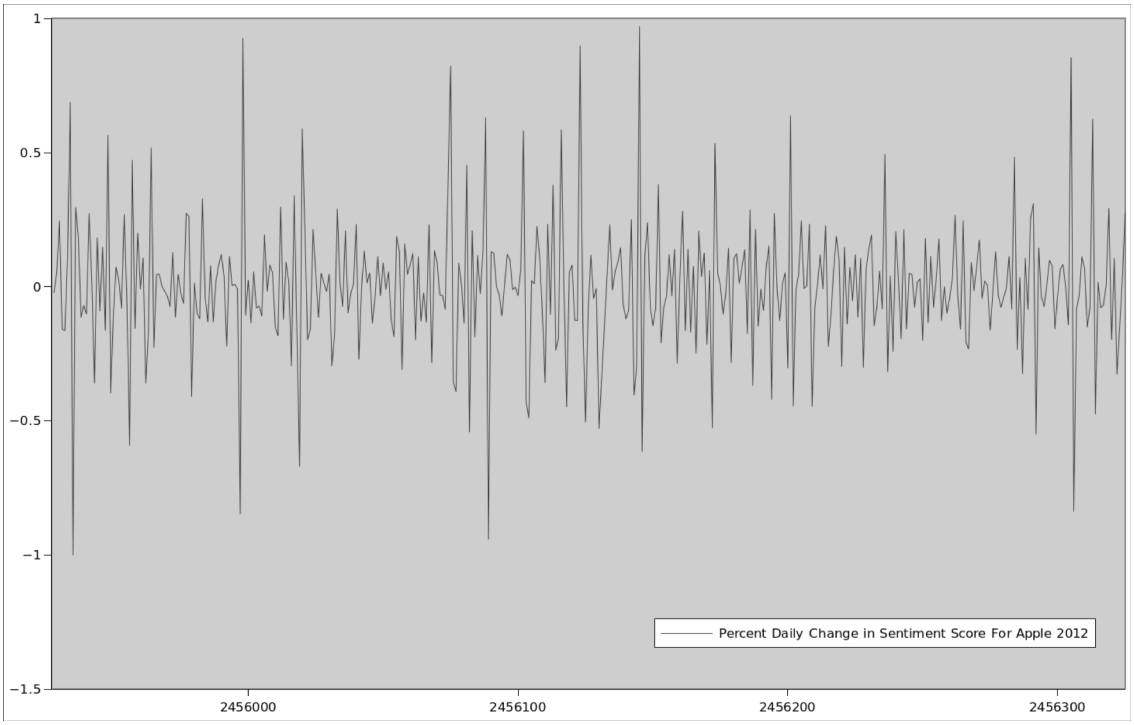

Figure 5: Sentiment Scores for Apple 2012

SENTIMENT ANALYSIS WITH ZAPHOD

The Zaphod utility is a naive sentiment scoring tool written in Java. Zaphod provides primitive scoring for basic sentiment analysis on a given topic, shown in Figure 4.

Zaphod reads in "positive" and "negative" word lists. It analyzes a provided block of input text, generating a score based on positive and negative word counts. Each sentence that is evaluated is counted as either positive or negative based on the preponderance of positive or negative words. Basic filtering by keyword is provided to eliminate sentences that do not mention a particular topic. Simple negation is also used to increase the accuracy of the results. For example, the phrase "not good" would be identified as a negative sentiment. The counts of positive and negative sentences are compiled for display to the user, and an overall score is generated.

Although Zaphod ultimately produces score files that link articles to positive and negative sentiment scores, results from Zaphod can also be examined on a sentence by sentence basis. This allows the user to see how Zaphod is grading each sentence and should indicate obvious problems, such as those discussed in the next section.

	B	C	D	E	F	G	H
1	Julian Day	Positive	Negative	Difference	Total	Percentage	Percent Change
2	2455927	10	6	4	16	0.25	
3	2455928	94	59	35	153	0.22875817	−0.0212418300654
4	2455929	50	28	22	78	0.28205128	0.0532931121166
5	2455930	84	26	58	110	0.52727273	0.2452214452214
6	2455931	89	41	48	130	0.36923077	−0.158041958042
7	2455932	41	27	14	68	0.20588235	−0.1633484162896
8	2455933	130	68	62	198	0.31313131	0.1072489601901
9	2455934	3	0	3	3	1	0.6868686868687
10	2455935	20	20	0	40	0	−1
11	2455936	57	31	26	88	0.29545455	0.2954545454545
12	2455937	125	44	81	169	0.47928994	0.1838353953739
13	2455938	101	47	54	148	0.36486486	−0.1144250759635
14	2455939	147	80	67	227	0.29515419	−0.0697106798428
15	2455940	80	54	26	134	0.19402985	−0.1011243342758
16	2455941	11	4	7	15	0.46666667	0.2726368159204
17	2455942	60	21	39	81	0.48148148	0.0148148148148
18	2455943	32	25	7	57	0.12280702	−0.3586744639376
19	2455944	62	33	29	95	0.30526316	0.1824561403509
20	2455945	65	42	23	107	0.21495327	−0.0903098868667
21	2455946	94	44	50	138	0.36231884	0.1473655695517
22	2455947	63	42	21	105	0.2	−0.1623188405797
23	2455948	15	2	13	17	0.76470588	0.5647058823529
24	2455949	13	6	7	19	0.36842105	−0.3962848297214
25	2455950	87	50	37	137	0.27007299	−0.0983480599308
26	2455951	137	67	70	204	0.34313725	0.0730642622012
27	2455952	244	113	131	357	0.36694678	0.0238095238095
28	2455953	155	86	69	241	0.28630705	−0.0806397247696
29	2455954	112	32	80	144	0.55555556	0.2692485016136
30	2455955	3	1	2	4	0.5	−0.0555555555556
31	2455956	5	6	−1	11	−0.09090909	−0.5909090909091
32	2455957	78	35	43	113	0.38053097	0.4714400643604
33	2455958	95	60	35	155	0.22580645	−0.1547245218384
34	2455959	87	35	52	122	0.42622951	0.2004230565838
35	2455960	112	46	66	158	0.41772152	−0.0085079892094

Figure 6: Sample of Sentiment Scores

RESULTS

Using the various tools described, news articles were retrieved for the year 2012 concerning the Apple Corporation, and processed to produce sentiment scores. Zaphod sentiment scores for Apple over the year 2012 can be seen in Figure 5. The scores are displayed as a percent change in sentiment from day to day.

A sample of sentiment scores produced by the tool is shown in Figure 6. Weekends are clearly demarcated from weekdays by a drastic reduction in the number of financial articles available. Sentiment is presented as number of articles with more positive sentences than negative (positive column), number of articles with more negative sentences than positive (negative column), and the difference between the two. A percentage is calculated of positive articles to total articles. Percent change from day to day is also shown.

NEXT STEPS AND PROJECT ROADMAP

The following section identifies improvements and future work planned for the project.

Google Custom Search: Expand CSE Using New Keywords Feature

When the current CSE was created, the websites to be included in the search had to be added manually. As such, the included websites were carefully selected for content type and subject matter and were limited in number. Google has recently added a feature whereby sites can instead be added using keyword groupings [10]. By making use of this feature, we will be able to quickly and easily increase the scope of the analysis tool by adding many more sites from which to aggregate data. This feature will also make it easier to create new CSEs for future work on other topics.

Zaphod: Subject Identification

The most prominent problem with the current approach is the lack of contextual identification by subject. The naive analysis only identifies whether a sentence is generally positive or negative and also mentions one of the provided keywords. The analysis does not determine that the sentence actually applies to the subject under examination. This approach can, for example, mistakenly identify positive sentences referring to another subject as a positive sentiment if the sentence mentions the current subject in any context. As an example, consider the following sentence regarding the Nokia Lumia, a competing product with the Apple iPhone. The sentence is intended as a positive statement regarding the Lumia, but mentions the iPhone as a counterpoint.

> *Plus the Lumia has many unique features that may convince some to switch over such as wireless charging, Nokia maps, free music, a new screen display that works with gloves on, and its Pure View camera which beat out the iPhone 5 and Galaxy Note II in most comparisons.*

The sentence is misidentified by Zaphod as a positive sentiment regarding Apple. Migrating to a more sophisticated textual analysis tool will allow more accurate subject identification and should alleviate this problem.

Zaphod: Specialized Vocabulary

The positive and negative word lists originally used are generalized word lists. They do not account for the specialized vocabulary used in the field under examination. Analysis performed with these general lists may mistakenly identify a word as having the wrong connotation, or fail to identify the sentiment associated with a word when the subject uses the word in a specialized context. As an example, consider the word "buy." In financial analysis, this word is used as a noun to indicate a good prospective stock purchase. Referring to a particular stock as a "buy" is a positive sentiment in the specialized vocabulary of business, but does not appear in the generic positive word list. A sentence containing this word might be identified as a sentiment-neutral statement, as in the following:

> *Apple is fundamentally and technically a buy at this level.*

The above sentence is identified by Zaphod as a sentiment-neutral sentence, when it is clearly a strong positive statement regarding Apple. In order to resolve this issue, a set of sentiment words peculiar to the financial industry

has been obtained and will be incorporated in all future analysis.

Demonstration of Correlation with Stock Price

In addition to the improvements identified above, future work is planned to demonstrate the accuracy of the toolset. The sentiment scores produced by the tool will be compared with the stock price of the same company over the same time period. Linear regression analysis will be used to demonstrate correlation. A moving average of the stock price, adjusted against the market composite for the particular industry, will be used as the dependent variable. By adjusting the absolute price against the market composite, we hope to remove spurious effects from more general market forces and thereby isolate the effect of sentiment on the stock movement. The sentiment scores produced by the tool will be treated as the independent variable.

CONCLUSIONS

In this paper, an automated platform for news aggregation and sentiment analysis was presented. The platform offers the user a high-level way to gather news articles over a given time frame concerning a specified topic and produce machine analyzable results that can be used for a variety of research tasks. The toolset is capable of acquiring large volumes of research data without a subscription to a news aggregation service. This will be a valuable boon to the researcher, who will be able to more quickly accumulate and analyze data on any given topic for analysis. Possible application areas include a wide array of topics, such as: financial analysis, marketing, epidemic monitoring, drug side-effect identification, and political opinion polling. Future steps include expanding search sources, incorporating more advanced textual analysis for improved subject recognition, integrating specialized vocabulary, and demonstrating correlation between sentiment and stock price.

REFERENCES

1. W Antweiler and M Frank, "Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards," *The Journal of Finance*, vol. 59, no. 3, pp. 1259-1294, 2004.
2. S Das and M Chen, "Yahoo! for Amazon: Sentiment Extraction from Small Talk on the Web," *Management Science*, vol. 53, no. 9, pp. 1375-1388, 2007.
3. A Devitt and K Ahmad, "Sentiment Polarity Identification in Financial News: A Cohesion-based Approach," *ACL*, June 2007.
4. R Feldman, "Techniques and Applications For Sentiment Analysis," *Communications of the ACM*, vol. 56, no. 4, pp. 82-89, 2013.
5. P Ferguson et al., "Exploring the Use of Paragraph-level Annotations for Sentiment Analysis of Financial Blogs," 2009.
6. N Godbole, M Srinivasaiah, and S Skiena, "Large-Scale Sentiment Analysis for News and Blogs," in *ICWSM*, vol. 7, 2007.
7. Google. (2013, October) Google Developers. [Online]. https://developers.google.com/custom-search/json-api/v1/overview
8. Google. (2013, September) Google Developers. [Online]. https://developers.google.com/custom-search/docs/structured_search
9. Google. (2013, September) Google Developers. [Online]. https://developers.google.com/custom-search/docs/structured_data#pagemaps
10. Google. Google Support. [Online]. https://support.google.com/customsearch/answer/3427771?hl=en&ctx=topic&topic=0000
11. H Kanayama and T Nasukawa, "Fully Automatic Lexicon Expansion For Domain-Oriented Sentiment Analysis," in *Conference on Empirical Methods in Natural Language Processing*, 2006, pp. 355-363.
12. B Liu, "Sentiment Analysis and Subjectivity," *Handbook of Natural Language Processing*, vol. 2, pp. 627-666, 2010.
13. B Pang and L Lee, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1-2, pp. 1-135, 2008.
14. B Pang, L Lee, and S Vaithyanathan, "Thumbs Up?: Sentiment Classification Using Machine Learning

Techniques," in *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing*, vol. 10, July 2002, pp. 79-86.

15. A Swartz. (2011, January) html2text. [Online]. http://www.aaronsw.com/2002/html2text/
16. I. H. Witten and F. Eibe, *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco: Morgan Kaufmann, 2005.