visualRSS: A Platform to Mine and Visualise Social Data from RSS Feeds

Martin O'Shea and Mark Levene

Department of Computer Science and Information Systems,
Birkbeck, University of London, United Kingdom
{martin,mark}@dcs.bbk.ac.uk

Abstract. RSS, a popular method of syndicating frequently updated on-line content, allows data to be stored in a semi-structured, XML-based format. Much work has been carried out applying data mining techniques to RSS, but in this paper we propose the visualRSS (vRSS) application as a platform to mine and visualise data trends in RSS feeds, by tracking changes in keyword frequencies as a source of social data. Core components of vRSS's architecture to manipulate RSS feeds are described. We also present the results of vRSS's initial experimental usage involving 36 students in late 2011, concerning our research into preferences of mining types and visualisations.

Keywords: RSS feeds, keyword frequencies, visualisations, social data, data mining.

1 Introduction

XML has become the *de facto* means of exchange [1] for transmission of data on-line, either in the form of documents or information exchanged between databases. RSS ('Really Simple Syndication'), a dialect of XML, provides a popular method of syndicating and aggregating on-line content, and most commonly consists of frequently updated works such as blog entries, news headlines, audio and video media, and HTML. Typically, a feed is composed of a `<channel>` containing the feed's title and description, and within the `<channel>` are numerous `<item>` elements, each of which forms a posting to the feed. In turn, each `<item>` is made up of `<title>`, `<description>` and publication date `<pubDate>` elements.

As described in this paper, much work has focused on applying data mining techniques to RSS feeds to classify and cluster them. But this work may be constrained by the semi-structured nature of RSS, volume of available data and the frequent inclusion of other, often unstructured, content. Despite this, it is the authors' hypothesis that RSS contains undiscovered information which may be beneficial to end-users.

In a previous case study [2], we presented the results of an experiment concerning the feasibility of mining and visualising textual and numeric information from the *raw* data of small numbers of RSS feeds. Our current work is underpinned by the successful mining of textual data from RSS in this case study.

M. Grossniklaus and M. Wimmer (Eds.): ICWE 2012 Workshops, LNCS 7703, pp. 121–133, 2012.

Moreover, we have extended this work to provide a variety of mining types to explore and visualise data trends in RSS by tracking changes in keyword frequencies. This is the basis of the visualRSS (vRSS) platform proposed in this paper, i.e. a research prototype to provide these services by integrating third-party products into a coherent and innovative toolset.

To allow it to be used by any class of user, vRSS employs several simple mining types for the specification of feeds and keywords. The application's outputs are a series of familiar visualisations including column and bar charts, treemap, pie chart and a wordcloud (sometimes known as a tag cloud), to display keyword frequencies as social data. By *social data*, we are not referring to data which represents user interaction within a social network such as Facebook or Twitter. Rather we define social data as actionable and potentially useful [3] information derived from datasets generated by social media [4], which may be relevant to anyone who cares to use it, e.g. to apply vRSS to feeds produced by news or financial sources, where the outputs are available for data warehousing, on-line trending in advertising and marketing, or in other big data analytics [5].

Therefore, this paper is written to describe vRSS and its use in the authors' research work. We begin by briefly discussing related work and we then describe vRSS and core components of it. We then summarise initial experimental use of vRSS together with the research aims and results of this work. We conclude by discussing our on-going work to classify RSS feeds by their keyword frequencies and analysing them for sentiment.

2 Related Work

Mining RSS falls within the scope of both textual and data mining. However, despite a massive corpus of available work in these areas, we focus explicitly upon RSS in the following brief literature review.

Thelwall et al. [6] distinguished between a *purist* mining of RSS as it is found, and a *pragmatic* use of extensive data cleansing. After using a purist approach to track stories in RSS feeds focusing on public fears about science, they concluded that, despite useful information in RSS, extensive and repetitive content requires data cleansing. This pragmatic approach has been more widely adopted in recent work clustering and classifying text from RSS feeds, of which [7], [8], [9] and [10] are examples. Roesler [11] has also identified caveats here concerning the number of documents or RSS feeds/items to be clustered, semantic and linguistic issues, and the time taken to cluster content especially in a real-time application.

Association rules have also been used to analyse news disseminated on the web: Hsu [12] has proposed the Web News Search System to discover 'useful' news, and Kittiphattanabawon and Theeramunkong [13] mined relations between Thai news articles concerning politics, economics and crime. A corollary of this concerns mining text snippets, e.g. the short RSS `<title>` element, which may not be sufficient for mining. Banerjee et al. [14] sought to improve the clustering of small pieces of text by supplementing their descriptions with text from Wikipedia. Phan et al. [15] again used Wikipedia and other sources to classify sparse text.

A detailed survey of methods available for the visualisation of text streams is given by Šilić and Bašić [16]. Wanner et al. [17] have visualised RSS data to reveal the sentiment of RSS news feed stories about the candidates during 2008's US presidential election campaign. This work is an example of the role of visualisation in social data analysis [18], i.e. to address the issues of whether visualisations enhance social networking and how users respond to them, how visualisations are used and the purposes they are used for. 10x10 [19] is an interactive exploration of the words and pictures in RSS feeds provided by several leading international news sources.

This role of visualisations though should not be confused with more typical social networking services like Facebook and MySpace. Instead, it is more related to data-centric social networking allowed by websites specialising in visualisations, e.g. Many-Eyes, allowing users to upload data, visualise it on-line and append comments.

3 visualRSS: Exploring and Visualising Trends in RSS

vRSS is a research prototype written to explore and visualise data trends in RSS feeds by tracking changes in keyword frequencies, where resulting social data is available to users via on-screen displays and interactive visualisations (Fig. 1). Users are able to specify feeds and keywords for mining via three simple mining types:

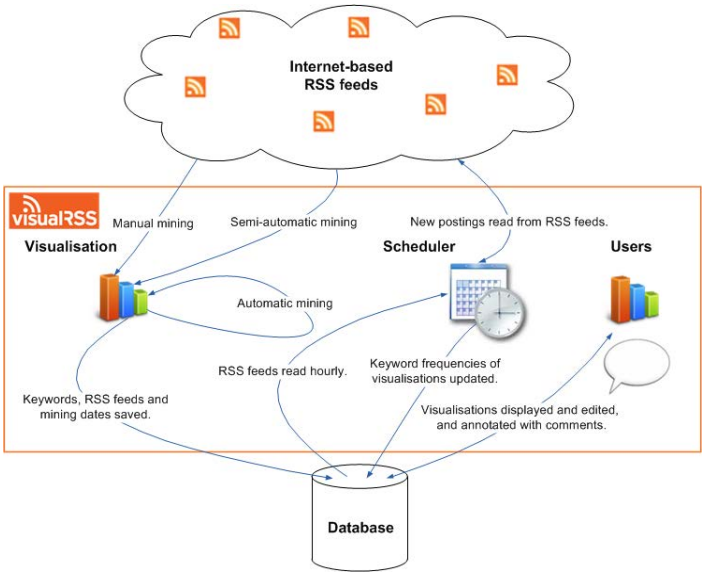

Fig. 1. A conceptual view of visualRSS

1. *Automatic mining* displays a current *buzz* of keywords in the *rssosphere*. It does this by using a subset of keywords mined hourly from the `<title>` elements of new postings to vRSS's pre-defined feeds: each keyword then has its frequency calculated from the `<description>` elements. Finally these frequencies are sorted in descending order, and the most popular keywords are determined as the subset.
2. *Semi-automatic mining* allows users to enter their own keywords, which search vRSS's pre-defined feeds to track topical issues.
3. *Manual mining* allows users to enter feeds and keywords of their own choice, to focus upon a particular subject(s).

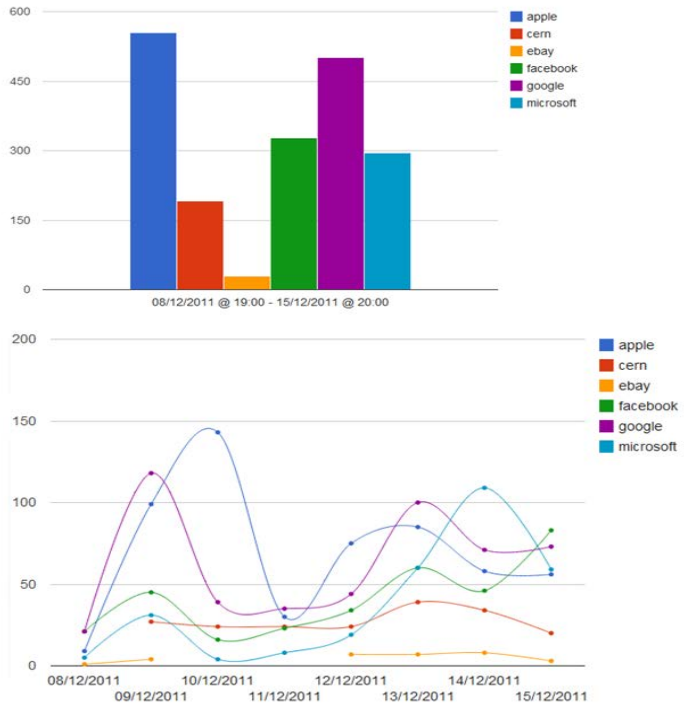

Fig. 2. A typical visualisation with aggregation (above), and time-series plot (below)

For purposes of uniformity, keywords in vRSS are currently simple English language unigrams without context or meaning. Pre-processing removes non-alphabet characters and stop words: all numbers are treated as positive. When they are defined using one of the mining types above, a wordcloud (or tag cloud) constantly displays current keyword frequencies from the appropriate feeds over the last 1 - 24 hours: see Fig. 3 for manual mining. Once a user is satisfied with

their selection, a sample visualisation is created via Google Chart Tools [1], i.e. a bar or column chart, pie chart, wordcloud or treemap, which is used as the basis of a new *permanent* visualisation. If saved, the sample visualisation's type, RSS feeds, keywords and dates between which mining will occur, are persisted in vRSS's database. If a new RSS feed is defined by a user during manual mining, a new database table is dynamically created.

A saved visualisation includes two charts, e.g. Fig. 2 displays several IT-related keywords for a week in December 2011: aggregated frequencies are displayed in the user-selected type, i.e. in this example a column chart, and the time-series chart depicts frequency changes during the aggregation's period. Each saved visualisation also displays its component feeds and keywords. A keyword-based search facility also allows users to browse vRSS's feeds and visualisations.

4 Architecture

4.1 Implementation

vRSS's basic architecture forms a typical *n*-tiered web application rendered in Java servlets and JSPs within an Apache Tomcat container based over a MySQL database. To implement the application, numerous third-party products and web services are used on a *black box* basis as a *mash-up*. Moreover, no frameworks tools such as Spring are used in vRSS: instead each principal object type, e.g. visualisations, has a dedicated class implementing methods for the necessary object relational modelling for database interaction. As each method handles one operation per table(s), e.g. add row, get one or many rows, such methods are quickly written and customised: each of these methods also makes use of a simple connection pooling.

4.2 Anatomy of a Mining Type

The three mining types in vRSS all employ the same basic interface to allow keywords and feeds to be specified. A wordcloud showing keyword frequencies from the appropriate feeds over the last 1 - 24 hours is displayed at the top of the page, and dedicated controls per mining type are placed adjacent to this, e.g. as Fig. 3 for manual mining.

Behind each mining type, a simple hierarchy of classes maintains the feeds and current frequencies of keywords specified: these are illustrated in Fig. 4. The super class of this hierarchy is an `RSS_Feed_Miner` which includes dedicated naming elements and an `RSS_Feed_Polling` object. RSS feeds are stored in a series of parallel lists along with the RSS elements, categories and the mining type to be used. The `RSS_Feed_Occurrence_Miner` specialisation class for keyword frequencies maintains a *key-value*, i.e. word-frequency, hashmap, and is populated from the wordcloud displayed in Fig. 3. Thus, the frequency of a particular keyword is derived from all of the feeds stored in the super class when a mining

[1] `https://developers.google.com/chart/`

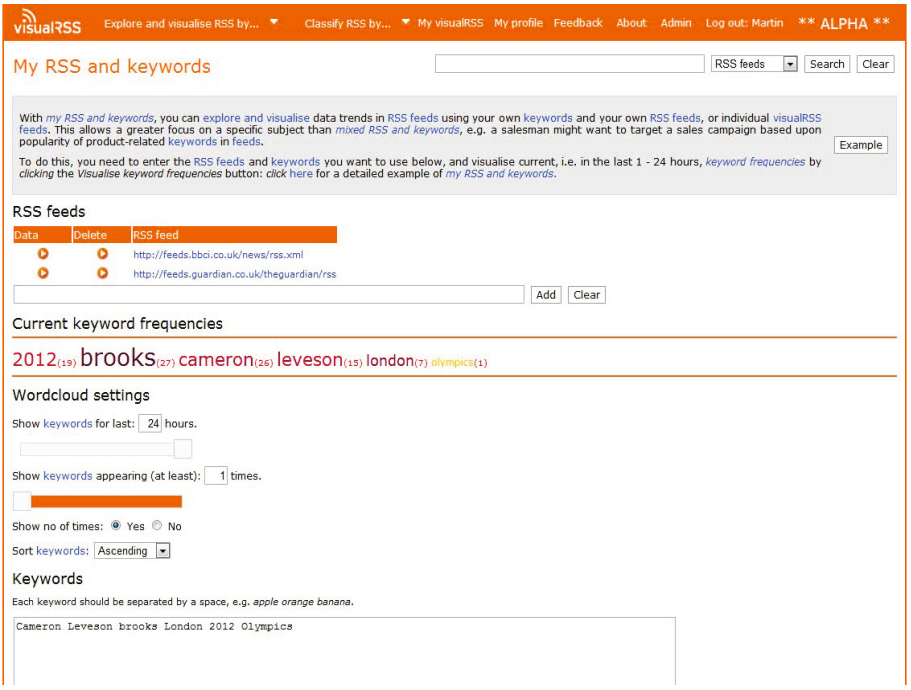

Fig. 3. A partial screenshot of manual mining in vRSS

type is used. The other specialisations illustrated are for future data mining of RSS feeds by vRSS which we describe later, and also for mining numeric data.

4.3 Polling and Indexing RSS Feeds

To maintain frequencies of keywords for visualising, vRSS relies on an index which is updated hourly with new postings mined from its pre-defined collection of RSS feeds. Thus, the index is structured as a series of M:N database relationships to record keyword frequencies from various `<item>` elements of RSS feeds on an hourly basis, i.e. Table 1 displays a simplified representation of the index.

The following pseudocode represents the basic hourly algorithm. The current polling date/time is determined (line 1) and two consecutive stages are executed: the first polls RSS feeds for new `<item>` elements whilst the second mines and disseminates this data to visualisations. Each stage is represented by the two `for` loops (lines 2 and 12):

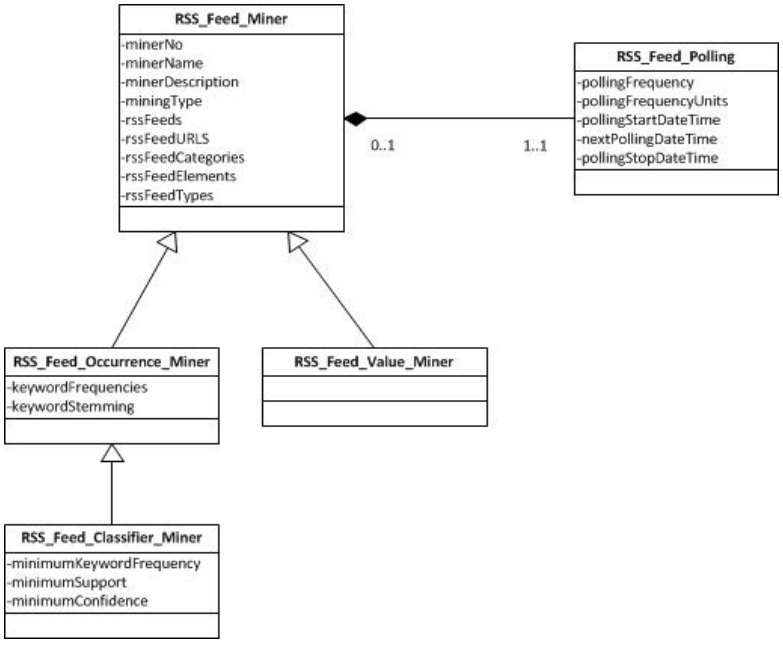

Fig. 4. vRSS's object hierarchy for storing RSS feeds and keywords

Table 1. A representation of the keyword index in vRSS

Polling date/time	Keyword	RSS feed	RSS Element	Frequency
29/04/2012 @ 11:00	$keyword_1$	$rssFeed_1$	`<title>`	4
29/04/2012 @ 11:00	$keyword_2$	$rssFeed_1$	`<description>`	2
29/04/2012 @ 12:00	$keyword_1$	$rssFeed_1$	`<title>`	6
29/04/2012 @ 12:00	$keyword_2$	$rssFeed_3$	`<description>`	1
29/04/2012 @ 12:00	$keyword_n$	$rssFeed_n$	`<description>`	3

The first of these loops (line 2) mines data from each RSS feed in vRSS: the address of each feed is polled (line 3) and the published date/time of each `<item>` in the feed is checked to determine if any new postings have been made during the polling date/time (line 5). For any new `<item>`, each `<element>` is parsed (lines 6 and 7), and new keywords are added to the index (lines 8 and 9): the frequency of each keyword per `<element>` is calculated in Lucene[2] and written to the index together with the feed, `<element>` and the polling date/time (lines 10 and 11). At the moment, only the frequencies of keywords from the `<description>` elements of each `<item>` are parsed when feeds are polled hourly. Postings from each feed are stored in dedicated database tables.

[2] `http://lucene.apache.org/`

The second `for` loop works per visualisation (line 12). For each keyword in a visualisation, its cumulative frequency for all of the visualisation's feeds, is retrieved from the index for the polling date/time (line 15): the visualisation is then updated (line 16) with the new frequencies. As before, keyword frequencies are visualised from only `<description>` elements of feeds.

```
1. set pollingDateTime = now - 1 hour
2. for each rssFeed
3.   poll rssFeed
4.   for each <item> in rssFeed
5.     if <pubDate> of <item> >= pollingDateTime
6.       for each for <element> in <item>
7.         parse <element> for keywords
8.         if new keyword found
9.           add keyword to index
10.        calculate frequency of keyword in element
11.        update index with pollingDateTime, keyword, freq., rssFeed and <element>
12. for each visualisation
13.   get keywords for visualisation
14.   for each keyword
15.     get frequency of keyword in rssFeed from index for pollingDateTime
16.   update visualisation with pollingDateTime, keywords and frequencies
```

4.4 Parsing an RSS Feed

During the polling process described above, vRSS uses the Rome API[3] to parse RSS feeds. Rome is based upon the JDOM XML parser and allows RSS and Atom to be parsed via a common `<SyndFeed>`, i.e. syndicated feed, model. The following Java extract from vRSS parses a feed's address:

```
URL url = new URL(rssFeedURL);
URLConnection urlConn = url.openConnection();
XmlReader reader = new XmlReader(urlConn);
SyndFeedInput input = new SyndFeedInput();
SyndFeed rssFeed = input.build(reader);
```

Each `<item>` in a feed is a `<SyndEntry>` and API calls access each `<element>`:

```
List<SyndEntry> rssFeedItems = rssFeed.getEntries();
for (SyndEntry rssFeedItem : rssFeedItems) {
    String title = rssFeedItem.getTitle();
    ...
}
```

[3] `http://java.net/projects/rome`

5 Initial Experimental Usage

5.1 Rationale and Objectives

Our previous case study [2] to mine and visualise data from RSS allowed both textual and numeric data. With vRSS though, we have suspended the mining of numeric data, e.g. exchange rate fluctuations from RSS, because of difficulties encountered by the experiment's participants. Instead, we have concentrated upon simplifying and extending our text mining by focusing on a variety of mining types to specify keywords and feeds to provide social data.

Therefore, to assess these techniques to the maximum possible real-world extent available to us at the time, an *alpha* version of vRSS was tested by 36 part-time MSc students of various employment and experience backgrounds, in a new experiment during late 2011. This has allowed us to research preferences and efficiencies of mining types and visualisations, distribution of categories of feeds visualised, and common usage of these amongst the mining types.

5.2 Discussion: An Analysis of Our Results

RSS Feeds and Categories. We provided our students with 57 RSS feeds arranged into seven generic categories, e.g. Business, Finance and Economics (BFE), Fashion, Celebrity and Lifestyle (FCL), Film, Music, News and Current Affairs (NCA), Science, Nature and Technology (SNT), and Sport. This corpus of feeds and categories were selected to be English language in content, global or regional rather than applicable to a specific country, and also to be wide-ranging and relevant in nature: the majority of feeds were present in the NCA (16) category and SNT (10) categories.

Keyword frequencies from each feed were recorded for 10 days prior to the experiment itself which lasted for a fortnight. We did not wish to bias our students in any way because we wanted to collect a wide variety of data for our research questions: therefore, the experiment was very *free* in format. Our students were able to choose keywords, add new feeds and to use the mining types without restriction: at the end of the experiment 202 feeds, including new categories such as Travel and Astronomy, were being mined hourly for new postings. The most popular feed categories were NCA with 52 feeds (25.74%), SNT with 39 feeds (19.30%), and Sport (31 feeds or 15.35%): least popular were Entertainment and Arts (EA) with 6 feeds (2.97%), and Travel with 5 feeds (2.48%). Figure 5 displays the distribution of mining types per feed categories with semi-automatic as favourite. Some 99 (73.33%) of visualisations covered two feed categories, whereas only five (3.70%) included all twelve categories.

Distribution of Visualisations. The 135 visualisations created by our students are displayed per RSS feed category in Fig. 6. The column chart was the most popular type with 107 instances (79.26%), despite alternatives such as word cloud and treemap, neither of which is reliant upon the association of words to specific colours to relate information.

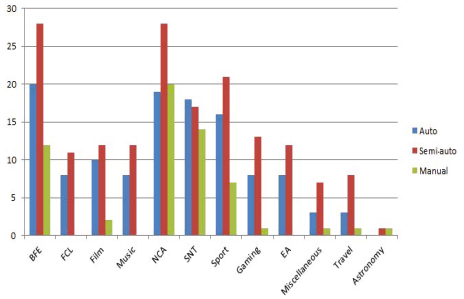

Fig. 5. Mining types per feed category

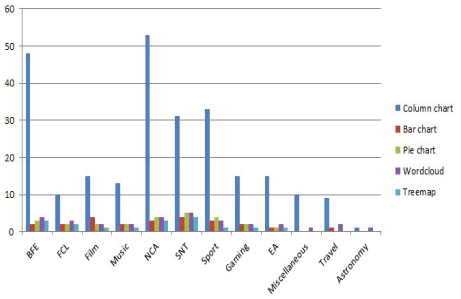

Fig. 6. Visualisations per feed category

Use of Mining Types to define RSS Feeds and Keywords. The majority of the 135 visualisations used different combinations of RSS feeds and keywords. But, in a small number of cases, students used the same feeds and keywords for semi-automatic and automatic mining: e.g. one student used keywords *economy*, *recession*, *depression*, *war* and *apocalypse* '*because of major events in current affairs*', where semi-automatic mining proved most successful because '*it tracked 4 keywords for 7 days*'. In this, and similar cases documented, automatic proved the least popular mining type because generic keywords convey '*less meaning and are less indicative of specifics*'. However, with automatic mining intended to provide a current *buzz*, this is not surprising.

6 Applications of visualRSS

In the experiment above, we also asked our students to propose applications for vRSS as a source of social data. Many of the suggestions made confirmed the authors' own opinions in areas such as:

– Business Intelligence: As a data source for big data analytics, or in turning unstructured data into tabular form for use in data mining fact and decision tables.

Fig. 7. vRSS as a web service

- Linguistics: To reveal geographical, cultural or political bias in news reporting, or calculating n-gram relationships between keywords to assist search engine results.
- Tracking and Trending: Where an organisation might place mouse-over adverts in web pages based upon popular keywords, or to track frequencies of keywords to determine market share.

More germane however, is the use of vRSS as a web service, i.e. as a browser plug-in or API to allow websites to display vRSSs outputs *on the fly* (Fig. 7).

7 Conclusion

In this paper, we have proposed the vRSS platform for exploring and visualising data trends in semi-structured RSS feeds by tracking changes in keyword frequencies. Major components of vRSS have been described and we have also presented a summary of our initial findings using vRSS. Though successful, this work was restricted to an experiment made up of a small user body in an ostensibly *class room* environment.

Our initial work also falls into the *purist* approach put forward by Thelwall et al. [6], where keywords are not extensively pre-processed in vRSS. This also contrasts with the approach taken by others, i.e. [14] and [15], in using external data sources to assist categorisation. vRSS differs from the related work we cite because it provides a coherent and innovative platform for aggregating information across RSS feeds. Furthermore, although vRSS's outputs are similar to others available, it must be remembered that RSS is a dialect of XML conforming to W3C standards, rather than a *proprietary* format belonging to a social network provider subject to the whims of a changing market.

Currently we are using decision trees to classify feeds into categories according to the presence of keywords at particular frequencies for varying 10, 20 and 30 day periods. This work re-uses the corpus of 57 feeds and seven categories described in our initial experimentation, and involves approximately 300,000 RSS feed `<item>` elements mined between August and October 2011. Further tests using Naive Bayes and SVM will also be carried out upon this corpus to compare the validity of the resulting classifications. We also plan to analyse our data for sentiment, and by relating the results to popular keywords revealed by our classification work, provide a finely-grained time-series analysis of RSS-based sentiment.

These extensions to vRSS extend our initial experimental work in providing social data from semi-structured RSS, which may be beneficial to end-users in the roles we have referred to.

Other future work also includes extending our current unigram keywords to include phrases and stemming.

References

1. Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., Yeargeau, F.: Extensible markup language (xml) 1.0, 3rd edn. W3C Recommendation (2004), http://www.w3.org/TR/2004/REC-xml-20040204/
2. O'Shea, M., Levene, M.: Mining and visualising information from RSS feeds: a case study. IJWIS 7(2), 105–129 (2011)
3. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann (2005)
4. Ohlhorst, F.: Tools to help analyze mountains of social data (2011), http://www.informationweek.com/thebrainyard/news/marketing/231002135/
5. Dumbill, E.: What is big data? An introduction to the big data landscape (2012), http://radar.oreilly.com/2012/01/what-is-big-data.html
6. Thelwall, M., Prabowo, R., Fairclough, R.: Are raw RSS feeds suitable for broad issue scanning? a science concern case study. J. Am. Soc. Inf. Sci. Technol. 57(12), 1644–1654 (2006)
7. Teng, Z., Liu, Y., Ren, F.: Create special domain news collections through summarization and classification. IEEJ Transactions on Electrical and Electronic Engineering 5, 56–61 (2010)
8. Getahun, F., Tekli, J., Chbeir, R., Viviani, M., Yetongnon, K.: Relating RSS News/Items. In: Gaedke, M., Grossniklaus, M., Díaz, O. (eds.) ICWE 2009. LNCS, vol. 5648, pp. 442–452. Springer, Heidelberg (2009)
9. Hu, C.L., Chou, C.K.: RSS watchdog: an instant event monitor on real online news streams. In: CIKM 2009: Proceeding of the 18th ACM Conference on Information and Knowledge Management, pp. 2097–2098. ACM, New York (2009)
10. Bossa, S., Fiumara, G., Provetti, A.: A lightweight architecture for RSS polling of arbitrary web sources. In: WOA (2006)
11. Roesler, R.: Relational RSS clustering techniques (2010), http://www.stanford.edu/class/cs229/proj2009/Roesler.pdf
12. Hsu, L.-F.: Mining on Terms Extraction from Web News. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) ICCCI 2010, Part I. LNCS, vol. 6421, pp. 188–194. Springer, Heidelberg (2010)
13. Kittiphattanabawon, N., Theeramunkong, T.: Relation Discovery from Thai News Articles Using Association Rule Mining. In: Chen, H., Yang, C.C., Chau, M., Li, S.-H. (eds.) PAISI 2009. LNCS, vol. 5477, pp. 118–129. Springer, Heidelberg (2009)
14. Banerjee, S., Ramanathan, K., Gupta, A.: Clustering short texts using wikipedia. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2007, pp. 787–788. ACM, New York (2007)
15. Phan, X.H., Nguyen, L.M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: Proceeding of the 17th International Conference on World Wide Web, pp. 91–100. ACM, New York (2008)
16. Šilić, A., Bašić, B.D.: Visualization of Text Streams: A Survey. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part II. LNCS, vol. 6277, pp. 31–43. Springer, Heidelberg (2010)

17. Wanner, F., Rohrdantz, C., Mansmann, F., Oelke, D., Keim, D.A.: Visual sentiment analysis of RSS news feeds featuring the US presidential election in 2008. In: IUI 2009 Workshop on Visual Interfaces to the Social and the Semantic Web, VISSW (2009), Online Proceedings, http://ceur-ws.org/Vol-443/paper7.pdf
18. Viégas, F.B., Wattenberg, M., Heer, J., Agrawala, M.: Social data analysis workshop. In: CHI 2008: CHI 2008: Extended Abstracts on Human Factors in Computing Systems, pp. 3977–3980. ACM, New York (2008)
19. 10x10 (2012), http://www.tenbyten.org/