EFFECTIVENESS OF RSS FEED ITEM DUPLICATION DETECTION USING WORD MATCHING

IAN K.T. TAN
Faculty of Information Technology
Multimedia University
Persiaran Multimedia, Cyberjaya
63100 Selangor Darul Ehsan, Malaysia
Tel: +60(3)83125235
Fax: +60(3)83125264
ian@mmu.edu.my

TZE-WEI SU
Faculty of Information Technology
Multimedia University

HAO-MING KHOR
Faculty of Information Technology
Multimedia University

EE-CHUN ONG
Faculty of Information Technology
Multimedia University

ABSTRACT

Users of feed aggregators know that duplicated articles are found occasionally on the feeds they subscribe to. It can be time consuming to read all articles and stumble upon duplicated items they have already read. Our work here is to determine the effectiveness of using basic word matching to remove duplicated items and only show the most relevant item, thus saving readers' time. The method described in this paper to remove duplicates involves word matching heuristics with an appropriate matching percentage. The duplicated feeds are then ranked to only display the highest ranked article. Ranking is done using the number of search items found on the titles of the news feeds where the highest number returned will be considered the highest ranked article. Using Malaysian online news feeds, our method found that with a matching percentage of 40%, our method will be able to minimize duplicates effectively with minimal errors. We did further empirical studies using 9 technology blog feeds over a longer period to provide us with a better averaging results. The matching percentage obtained is also within the same quantum. The method described here has a low overhead in terms of processing for the duplicates and with careful selection of matching percentage, the system will effectively remove the majority of duplicates.

Keywords: duplicate removal, word matching, RSS feed, news, ranking

INTRODUCTION

Early Internet users, who subscribe to many mailing lists, were usually swamped with tens or sometimes hundreds of emails every day. The subscriptions can be organized by applying filtering logic to place different email subscriptions into different folders. However, this becomes quite cumbersome and with the Internet being the main source of information for many users, the Really Simple Syndication (RSS) format became popular for the delivery of much of today's web content. According to a report by Universal McCann (Smith, 2008), the number of Internet users in 2008 who subscribe to RSS feeds is just slightly below 40% of the Internet user population, and 59% of those surveyed do read online news or blogs.

Most of the news and blogs available online will provide RSS feeds for their audiences, who can then use a RSS Reader or also known as RSS Aggregator to access them. There are numerous RSS Readers such as Feed on Feeds (www.feedonfeeds.com), RNews (www.rnews.com), Drupal Aggregator Module (www.drupal.com), NewzCrawler (www.newzcrawler.com), Google Reader (reader.google.com), FeedDemon (www.feeddemon.com), and SharpReader (www.sharpreader.net). Each of these feed aggregators has its own weaknesses and strengths. There are desktop-based aggregators and also web-based aggregators. Web-based aggregators have an edge over desktop aggregators because they can update the feeds even when the user's computer is not online. Besides, users can read their feeds from anywhere with just a browser. We summarized a few web-based feed aggregators, namely Feed on Feeds (Minutillo, 2009), RNews Feed Aggregator (Rollet & Wood, 2009), and the Drupal Aggregator Module (Bryon et al., 2008).

- Feed on Feeds (Minutillo, 2007) is a server side RSS feed aggregator written in PHP. The installation process is uncomplicated without much configuration requirements other than the setting of the administrator account and password. Feed on Feeds is an open source software that is distributed under the GNU Public License (GPL). It utilizes the SimplePie API (Parman & Sneddon, 2008) for most of the feed aggregation processes. Overall, Feed on Feeds is a light-weight feed aggregator whose database contains only six tables. It has basic features such as adding feeds, updating feeds, removing feeds and marking the feeds as favourites. The source code can be obtained from http://code.google.com/p/feed-on-feeds/. Feed on Feeds does not have any sorting nor duplication removal capabilities.
- Rnews (Rollet & Wood, 2009) is another feed aggregator written in PHP that is more feature-rich in terms of functions as well as the interface. It provides a multitude of views for the users to choose from, which includes block view, wide-block view and list views. The feeds are arranged in an organized manner where users are also able to categorize their feeds. On the functions available in Rnews, it provides for marking feeds as favourites, marking items as read or unread, and filtering and sorting feed articles. Similar to Feed on Feeds, Rnews utilizes a third-party code library for aggregating feeds. Rnews uses the MagpieRSS (Elliot-McCrea, 2011) instead of the SimplePie API. Rnews is more comprehensive than Feed on Feeds, and it provides a scoring feature which basically ranks the popularity of an article. This is achieved by computing the number of users who click on the link. However, the scoring is an overall scoring of all articles as it does not have a duplication detection or removal feature.

- Drupal (Bryon et al., 2008) is not a dedicated feed aggregator but a comprehensive content management system (CMS) that is written in PHP. However, it provides a feed aggregator module with its default installation. Similar to Feed on Feeds, it is a simple feed aggregator with limited features. It is capable of gathering and parsing feeds in RSS and the frequency of updating can be set accordingly. It supports categorizing of feeds, filtering of keywords and has different access controls for different users. Although it is able to categorize and filter keywords, it does not have the capability to rank the importance of an article nor does it assist in removing duplicated articles.

The other popular RSS Readers are client based RSS readers such as NewzCrawler (www.newzcrawler.com), FeedDemon (www.feeddemon.com) and SharpReader (www.sharpreader.net). These readers provide certain capabilities that are not found on web-based clients, such as the ability to automatically download music files to your portable personal music player or the ability to gather content from various other sources such as Usernet Newsgroups (NNTP) onto a single client. The client based RSS readers also have the advantage of a faster startup time, as the RSS articles can be downloaded in the background.

As none of the web-based clients or client based readers provide for duplicated article detection, ranking and removal, this often leads to users having to read duplicated items which are unproductive. Online news feeds, such as The Star Online (www.thestar.com.my) and News Strait Times Online (www.nst.com.my), will definitely be reporting similar news daily. In most cases, they will have different titles for the same event. This project aims to remove duplicated items, rank the articles and remove all duplicates except the highest ranked article. This will assist users to read only the most relevant articles.

We propose an efficient method to detect duplicates by using word matching where we compute the percentage of similar words used in different articles and through using a single matching percentage threshold, we identify them as duplicates or otherwise.

After identifying duplicated items, a further problem is posed; which of the duplicated article should be shown to the user? To resolve the problem, we ranked the articles based on the number of search items returned using a search engine on the articles' titles. The article's title with the most number of search returns will be used to display to the user.

Our contributions here are therefore;

- The identification of the appropriate word matching percentage for Malaysian online news feeds to identify duplicated articles.
- The identification of the appropriate word matching percentage for top technology blogs to identify duplicated articles.
- A news article ranking method using the number of search items returned by search engine on the articles' titles.

BACKGROUND

The problem of detecting digitally stored document duplicates has drawn of much interest since the 90s, when efforts to digitize content were the trend. Researchers such as Lopresti

(1999) have classified documents into four models, which are full or partial content duplication and whether the layout was maintained or not. The interest of his work then was to understand the effects of measuring duplication based on the four models for documents that were digitized and with focus on uncorrected documents that went through the Optical Character Recognition (OCR) process as opposed to printed and scanned documents.

In the area of using words that are deemed statistically important, the article by Hirao et. al. (Hirao, 2004) proposed the use of word statistics to apply to news articles. This combined with Adam et. al. (2008) can be used to identify titles that would contain similar content. However, the computation requirements will be prohibitive and will not be suitable for a simple heuristics that we are looking for. This is generally the issue with most natural language processing solutions where computational overheads may be a hindrance for our intent.

In the era of the Internet, the need to compare web sites and pages to detect plagiarism or repeated content became important. With research focus on search engine results, Li et al. (2005) proposed an algorithm based on the sequence of keywords, called the Keyword Sequence Method (KSM). Their method comprises a keyword extraction process, building the keyword sequence, and based on the proposed KSM, reduces false positives (wrongly detected document duplication). The KSM algorithm is resilient to document noise, which is additional text that is added to the original documents.

Other novel methods of detecting web news duplicate include the use of one way encryption algorithm, MD5, to produce the document signature (Wang et al., 2007) and the use of Fourier Transforms (Chen et al., 2008). The work done by Wang et al. (2007) involves chunking and segmenting of the documents before producing the respective MD5 fingerprints after removing extra whitespaces and punctuations. These fingerprints are then compared and if it matches, a near duplicate is considered to be detected. Chen et al. (2008) represented each webpage as several Fourier coefficients, and compared the Fourier coefficients to determine webpage duplication.

The proposed methods by Li et al. (2005), Wang et al. (2007) and Chen et al. (2008) require significant computing resources as either the algorithm complexity is $O(n^2)$ (Li et al., 2005) or the pre-processing requirements to produce the MD5 fingerprint or the Fourier coefficients is significant even before the actual comparison stage. There has been sufficient interest in trying to determine duplicates or near duplicate of web pages on a large scale. A comparison was made by Henzinger (2004), where she evaluated algorithms by Broder et al. (1997) and Charikar (2002), which were developed or used by search engine companies. Her empirical studies involved 1.6 billion distinct web pages. Wang and Liu (2009) also attempted to find duplicates in large scale web content. They proposed a duplication removal scheme that is based on the temporal vector of the article coupled with the feature codes of the article. Although this works well for large scale web content; usage of the temporal vector of an article is not relevant if we are considering duplicated news for the same day or within a day or two. In other words, all contents of interest to us will be in the same period.

The pre-processing stage used by Wang et al. (2007) and Broder et al. (1997) involves the removal of white spaces and punctuations, which we will also adopt and using similar ideas from Li et al. (2005), we propose to remove common words which will result in a set of words that are similar to the keywords extraction scheme by Li et al. (2005). Further to that, our pre-processing will involve the removal of repeated words, which is the reverse of the content summary efforts by Takeda and Takasu (2008) who proposed the use of specific

phrase frequency for document summary. Our work uses this concept of matching words in order to determine duplication instead of summarization; with the intent to expand the work to also include weightage for word sequences.

IMPLEMENTATION

In this work, we developed a prototype using the SimplePie API (Parman & Sneddon, 2008) for fetching and extracting RSS data from the feeds the user subscribes to. We use the API to extract data like titles, content and the date the article was posted. These data are then stored in a MySQL database for display to the user.

Pre-Processing RSS Feeds And Word Matching

For the duplication section, we compare the words contained in the content of the articles. Titles are not appropriate in the journalism industry as titles for similar content may differ widely with no syntactic or semantic relevancy between the different articles of a similar content. Furthermore, two different articles may have similar or even the same title, and they should not be identified as duplicates.

The first phase in removing duplicates is to remove all punctuation symbols or marks like commas, full stops, semicolons, and question marks. This is followed by the removal of all common words such as "the" and "an". This is to increase efficiency and accuracy of the system as it does not need to compare too many words and also reduces the possibility of wrongly identifying duplicates because of the common words both articles have. Our database of common words were deduced from the Internet and used for this process. It is by no means a comprehensive or accurate database of common words as it does not take the Malaysian usage of common English words into account. Once the data have been massaged, the word matching is conducted.

The words in one article are stored into one array and the words in the other article are stored in another array. We then used the PHP's built-in function (`array_intersect()`) to get a new array of words from both of the articles that matched. We then count the number of elements in this new array and divide it with the total number of words from both articles. After that, we calculate a percentage and identify duplicates through a percentage threshold. Only if duplication is detected will the articles be added to our database of duplicated articles. Below is a summary of our duplication method;

1. Initialize arrays to store articles to be compared and load common word database.
2. Pre-process the articles by:
    - Removal of HTML tags
    - Converting all alphabetic content to lower case
    - Removal of punctuation symbols or marks
    - White space replacement, and
    - Removal of common words.
3. Remove duplicated words within the arrays that store the articles.
4. Execute `array_intersect()` for both article arrays.

5. Execute `count_array_intersect()` to obtain the number of duplicates.
6. Compute using `match_percentage()` where the matching percentage is equal to

$$\frac{(\ \texttt{count\_array\_intersect()}\ \times\ 2\ )\ \times\ 100}{(\ \texttt{sum of both array count}\ )}$$

7. If the percentage is greater than the threshold set, return match, or else return no match.

Ranking

In obtaining duplicates, there is a need to determine the appropriate single news item that will be displayed. There are many proposals on ranking an article, such as work done by Svore et. al. (2007) that uses learning neural networks combined with Wikipedia entries in order to determine the most important article.

In our prototype, we loosely define ranking as search engine visibility where the importance of the content is indicated by the number of references to the article or site. As such, we will use the number of search engine returns as our ranking system where more returns will result in a higher rank. The highest ranking article will be displayed on the system interface. The lower ranked articles are not discarded but instead will be hidden from view and users can still access them through an icon.

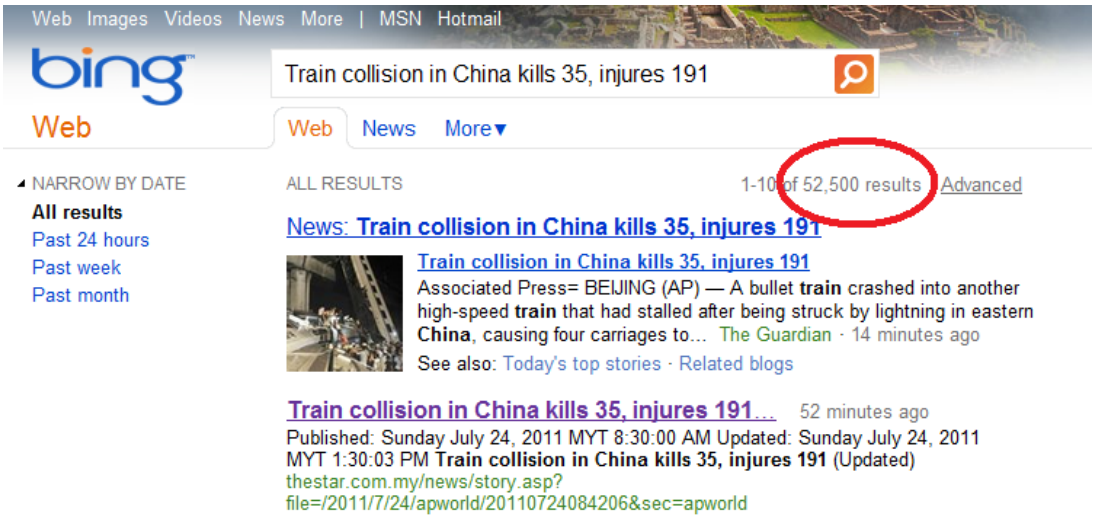

Figure 1. Viewing of Duplicates

Our method uses only the title of the article. This is because if we use the content as the search criterion, the number of search hits may be too large. In our work, we utilized the Microsoft Bing search engine. Upon obtaining a list of all duplicated articles, the titles of these duplicated items are searched using an automated script, where it will strip out all

content returned and locate the number of results returned as circled in Figure 1. The title that returns the highest results will be recorded as the highest ranked.

RESULTS

We did empirical studies using five news feeds and then reduced the number of news feeds to three. The feeds were selected as they formed the main stream news as well as alternative online news for Malaysia. We note that there are other online news feeds but we limited our initial studies to just the following;

- The Malaysian Insider – http://feeds.feedburner.com/tmi/news/malaysia?format=xml
- News Straits Times – http://www.nst.com.my/rss/sec?section=national
- The Star: Nation – http://thestar.com.my/rss/nation.xml
- Google News: Malaysia – http://news.google.com/news?ned=en_my&hl=en&topic=n&output=rss
- Bernama – http://web8.bernama.com/bernama/v5/rss/english.php

Data were collected over a period of 10 days and we recorded the number of items, number of duplicates found and manually determined the number of false positives as well as the number of false negatives. False positives are articles that are wrongly identified as duplicates and false negatives are articles that the system did not manage to identify as duplicates. The testing was conducted over a range of word matching percentages (threshold values).

Five Feeds

Figure 2 shows the results from taking the average number of false positives and false negatives over a period of 10 days for all five news feeds. We can note that there is a compromise between trying to reduce the number of false positives with an increase in false negatives.

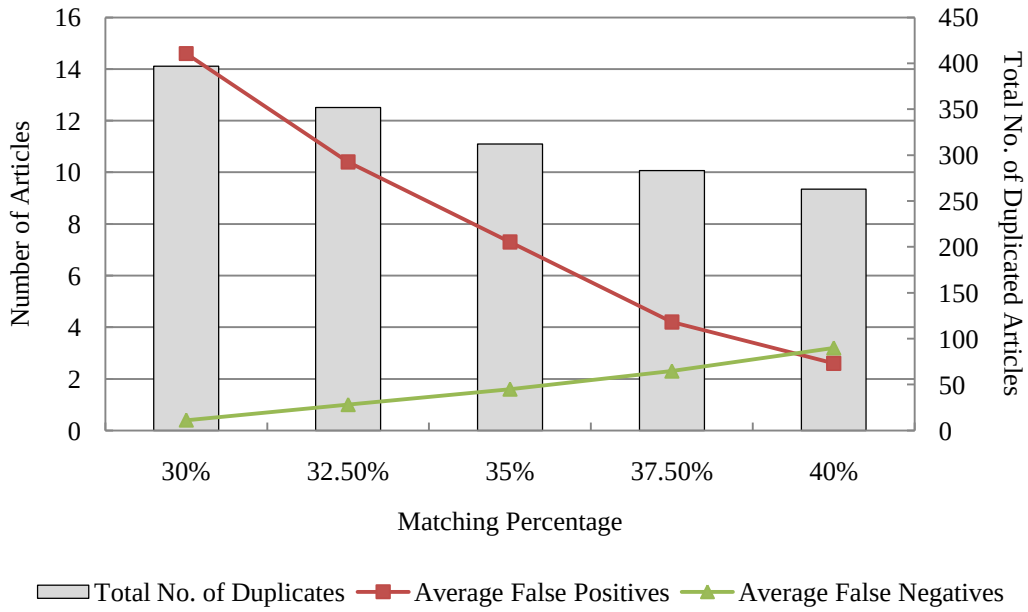

Figure 2. Five Malaysian News Feeds over 10 Consecutive Days

The number of false positives has to be reduced significantly as we will not want the users to miss out on any news whilst false negatives are not as critical since the role of the system is to help reduce the number of articles as opposed to completely eliminating duplicates. Averaging of the number of false positives and false negatives indicates that there is a need to set the threshold to greater than 40% in order to minimize the number of false positives.

We further analyzed our results by breaking them into the first five days and the last five days to determine the possible threshold value that is needed. Analyzing five feeds over a period of five days, it is noted (Figure 3) that the average number of false positives is less than the number of false negatives at a lower threshold than over a span of 10 days.

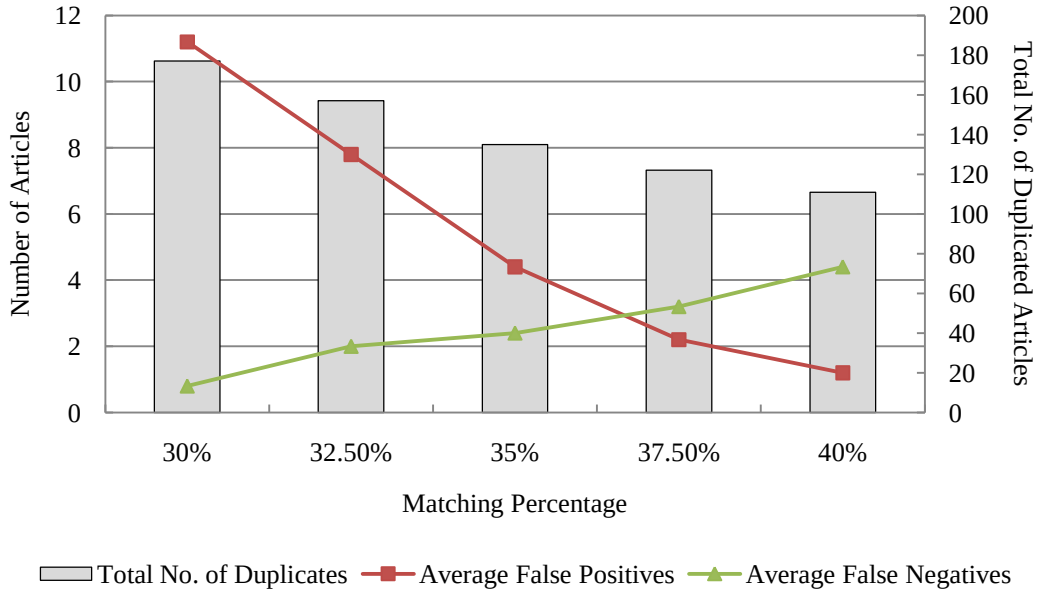

Figure 3. Five Malaysian News Feeds over "First" Five Consecutive Days

Figure 4 illustrates the Last 5 days where the number of false positives is significant whilst the number of false negatives is reduced. From our findings with the limited data set that we have gathered, this method of determining the threshold values is at most a coarse estimation.

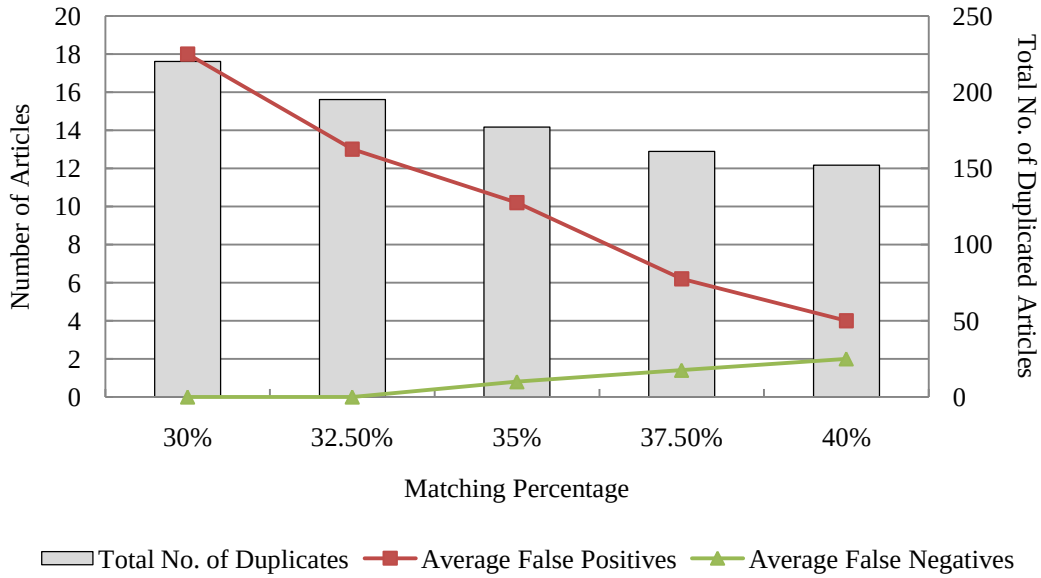

Figure 4. Five Malaysian News Feeds over "Last" Five Consecutive Days

Three Feeds

We tested the method using three news feeds (Malaysian Insider, News Straits Times and The Star Online) instead of five and we found that the number of false positives and the number of false negatives were reduced (Figure 5). This is because the number of articles to compare is reduced and hence false readings will also be reduced. It is also noted that similar to using five news feeds, the trade-off between reducing the number of false positives has an impact on the number of false negatives. A significant drop in the number of false positives can be seen at the 37.5% threshold with minimal (comparatively) increase of false negatives.

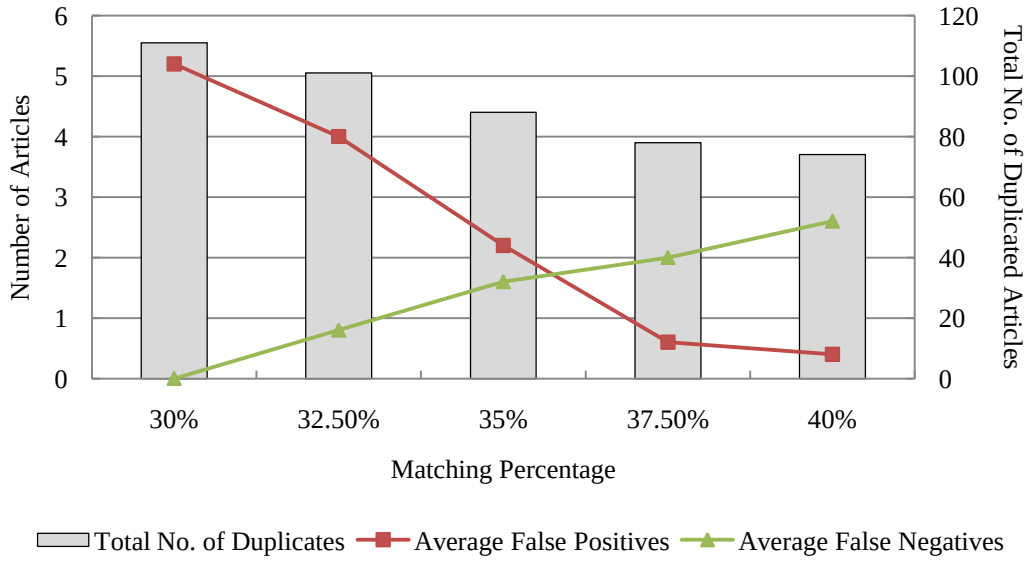

Figure 5. Three Malaysian News Feeds over 10 Consecutive Days

Similar to our analysis for the five feeds, we further analyzed our results by breaking them into the first five days and the last five days. Using just three feeds over a period of five days, the number of false positives (which we need to reduce as much as possible) is on average reasonably low when the threshold value is set to 37.5% and beyond (Figure 6 and Figure 7). This augurs well as the number of false negatives does not increase significantly in comparison.

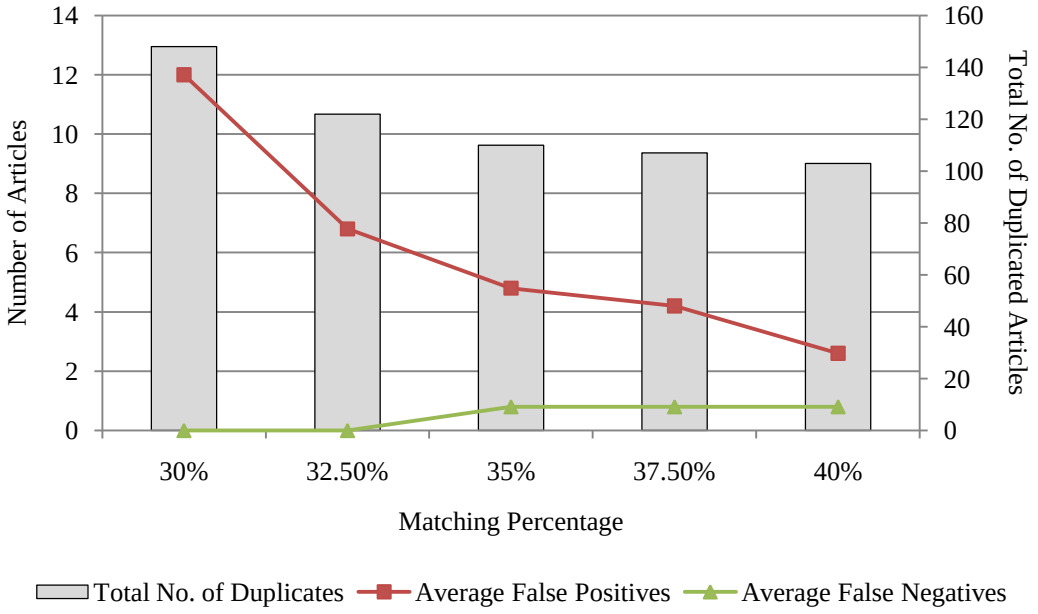

Figure 6. Three Malaysian News Feeds over "First" 5 Consecutive Days

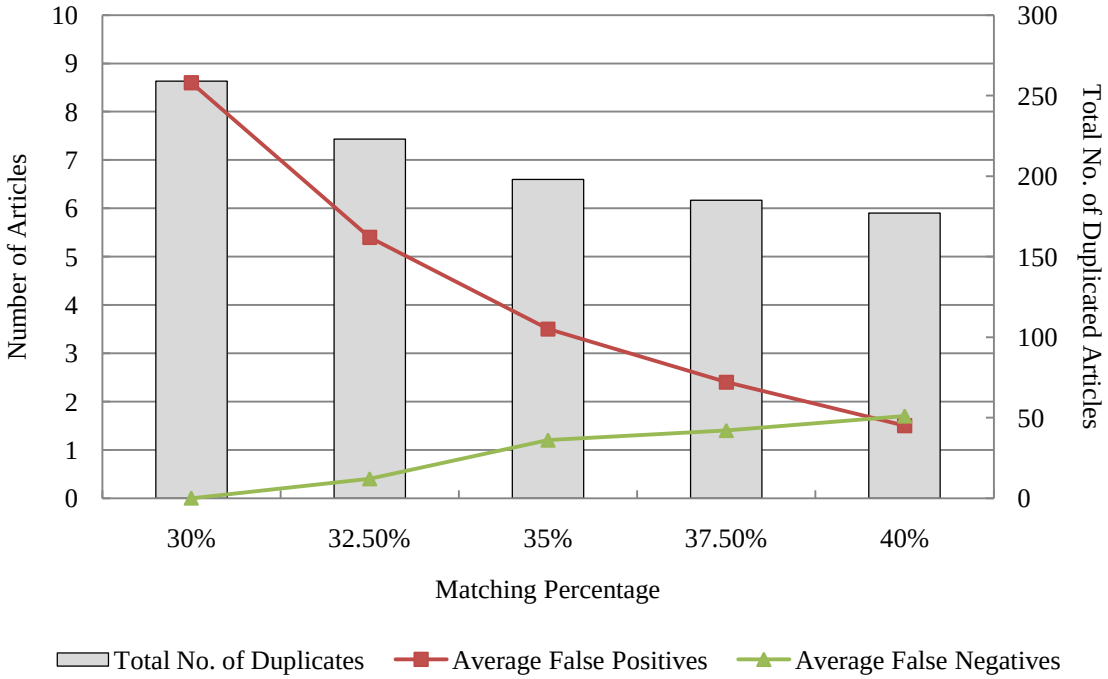

Figure 7. Three Malaysian News Feeds over "Last" 5 Consecutive Days

Based on Figures 2 – 7, the results indicate that a useable match percentage is greater than 40% in order to significantly reduce the number of false positives. It is noted that by increasing the threshold, the number of false negatives will also increase.

Another determining factor is the number of news feeds. Naturally, with a smaller number of feeds, this method of word matching will provide a better solution. This is because fewer news feeds means fewer number of items to analyse, thus less occurrence of wrongly identifying duplicates.

Technology Blog Feeds

As a comparison for our Malaysian News Feeds, we conducted the same empirical study for nine different technology blogs. From our previous results, we are already aware that the number of feeds will mean that there is a need to increase the threshold in order to minimize false positives. However, our hypothesis is that technology blogs will not have as many duplicated news articles, as each of the blogs will have a slightly different focus or different viewpoint on the different technologies; and they do not necessarily report similar news on the same day (as opposed to news feeds). To obtain a better averaging, we collected data over a longer period of 30 consecutive days. We performed the test using the following nine technology blogs;

- TechCrunch – http://feeds.feedburner.com/TechCrunch
- Gizmodo – http://feeds.gawker.com/gizmodo/full
- TheRegister – http://www.theregister.co.uk/headlines.atom
- TechDirt – http://feeds.techdirt.com/techdirt/feed
- GigaOM – http://feeds.feedburner.com/ommalik
- BetaNews – http://feeds.betanews.com/bn
- New York Times – http://feeds.nytimes.com/nyt/rss/Technology
- TheStar Technology – http://thestar.com.my/rss/it_news.xml
- Google Science Or Technology – http://news.google.com/news?pz=1&cf=all&ned=en_my&hl=en&topic=t&output=rss

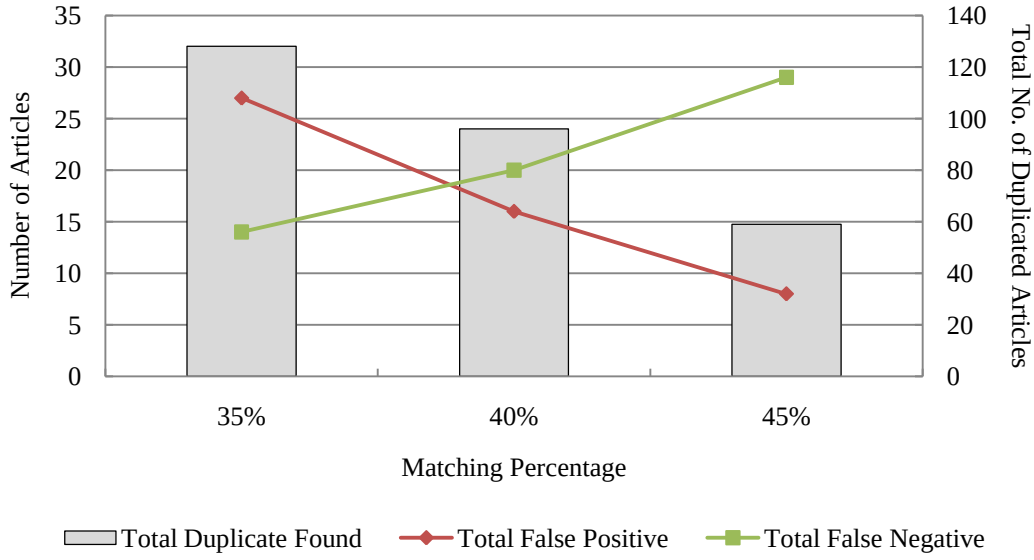

Figure 8. Nine Technology Blog Feeds over 30 Consecutive Days

Figure 8 depicts the results from our empirical study on using nine technology blogs over a period of 30 days. Note that in Figure 8, the number of false positives and the number of false negatives is not averaged out over the period of 30 days; they are the total numbers. It is hence noted that on average the number of false positives and false negatives is below 1.

FUTURE WORK

We have so far used the number of Search Engine results to rank the importance of news articles. This method is not suitable for our needs as the performance of obtaining the results will be counterproductive to our objective of finding an efficient method to identify duplicates, rank them and display the highest ranked article. The performance of obtaining the search results is highly dependent on the network connectivity and also the load on a third-party search service.

For future work on this, we propose the use of a language checker such as After the Deadline (AtD) (www.afterthedeadline.com) where the spelling, writing style and grammar can be checked. The sources for After the Deadline are distributed under the GNU Public License and can be downloaded from http://open.afterthedeadline.com.

We did preliminary investigation using the number of grammar and spelling errors returned by AtD as the measure of how well an article is written. The article with the lowest number of errors returned will be ranked highest. Note that the lowest value return by AtD is -2, which indicates that there is an error during the checking. The next lowest value is -1, which is used to indicate that there are no errors found. Otherwise, it will return the number of errors found. In our implementation, we ignore the article if it returns -2. In the event that

there are articles that share the same lowest return value, our system picks the first article we tested.

Instead of installing our own server, which will eliminate the drawbacks of dependency on network connectivity performance and the load on a third-party service, we used AtD's web service to do our preliminary study. We note that as part of the future work, we will install our own AtD server.

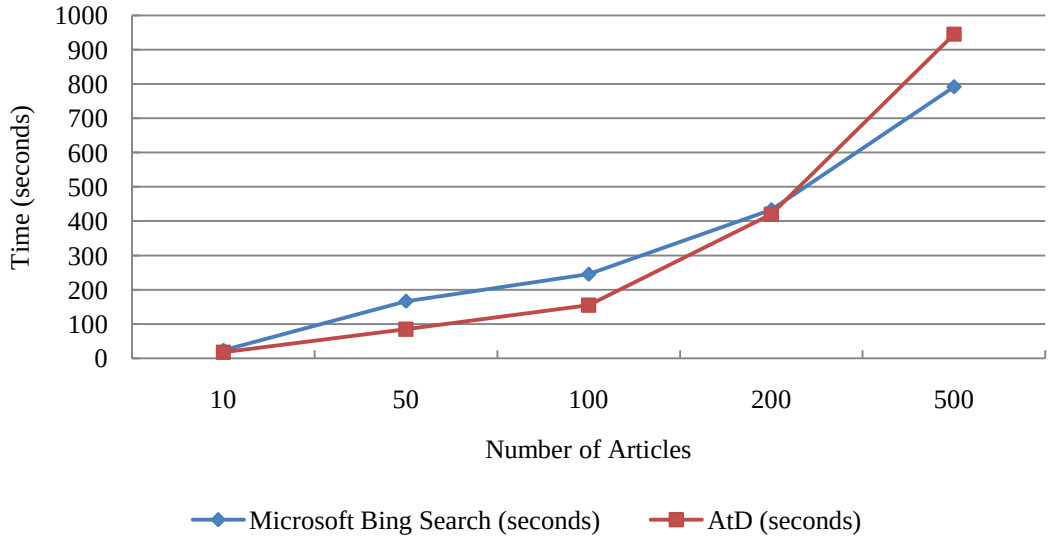

Figure 9. Performance Comparison between Using AtD vs Search Engine

We perform a simple measurement to give us a gauge of the potential performance of using the AtD. Figure 9 shows that the performance of using AtD as a web service over the Internet is not dissimilar to that of using the Search Engine and it shows improvement for up to 200 articles being tested. This is just to provide an indicative feedback on the potential performance. We note that in all probability, the Search Engine server is of a much higher specification than that of the AtD server and the network overheads are likely to contribute significantly to the overall performance measurement. With this preliminary work, we intend to pursue investigating the usage of the AtD checking capabilities as a local service instead of using other ranking methods such as tracking the number of clicks on the article by Elliot-McCrea, K. (2011).

CONCLUSION

From our results on the nine technology blogs (Figure 8), the average number of false positives falls below 1 when the threshold is set at 45%. This would mean that there will be days when there are no false positives and hence it would mean that no articles are wrongly

detected as duplicates. Users of our prototype system will hence not be missing any news articles and will benefit from the removal of some duplicated articles, hence saving time.

Our results gave a good indication that a basic word matching method can effectively achieve the task of removing duplicates with a careful selection of threshold values for the match percentage. It has shown to be effective in removing duplicates for Malaysian news feeds and even more effective if used on technology based RSS news feeds. There are strong indications that a higher threshold has to be set as more feeds are put into consideration but then we are also conscious that typical news feed subscribers will generally not subscribe to too many sites and would have their own list of favourite sites to monitor. This shows that our method is effective for such RSS Feeds.

We note that the current ranking method used is very primitive and the performance is highly dependent on external variables such as the Internet backbone and the number of articles being searched. For our future work, we propose the use of language quality checks instead of other methods that require user input or external services.

This work can be used to eliminate false positives efficiently and coupled with other pre-processing heuristics; false negatives can be further minimized. We also note that a simple word matching method will not be able to accurately remove duplication of news articles but it is an efficient method to reduce the number of duplicates.

REFERENCES

Adam, C., Delpech, E., & Saint-Dizier, P. (2008). Identifying and indexing titles in web texts. In D. C. A. Bulterman, L. F. G. Soares & M. G. C. Pimentel (Eds.), *DocEng'08: proceedings of the Eighth ACM Symposium on Document Engineering* (pp. 213-216). New York: ACM Press.

Broder, A. Z., Glassman, S. C., Manasse, M. S., & Zweig G. (1997). Syntactic clustering of the Web. *Computer Networks and ISDN Systems, 29* (8-13), 1157–1166.

Bryon, A., Berry, A., Haug, N., Eaton, J., Walker, J., & Robbins, J. (2008). *Using Drupal*. Sebastopol, CA: O'Reilly Media.

Charikar, M. S. (2002). Similarity estimation techniques from rounding algorithms, In J. H. Reif (Ed.), *STOC '02 Proceedings of the 34th Annual ACM Symposium on Theory of Computing* (pp. 380–388). New York: ACM Press.

Chen, J. Y., Sun J. Z., & Zhang Y.P. (2008). Finding near replicas of Web pages based on Fourier transform, *Journal of Computer Applications, 28*(4), 948–950.

Elliot-McCrea, K. (2011). MagpieRSS: RSS for PHP. Retrieved from magpierss.sourceforge.net on July 12, 2011.

Henzinger, M. (2006). Finding near-duplicate web pages: a large-scale evaluation of algorithms. In S. Dumais, E. N. Efthimiadis, D. Hawking & K. Jarvelin (Eds.), *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 284–291). New York: ACM Press.

Hirao, T., Okumura, M., Fukushima, T., & Nanba, H. (2004). Text summarization challenge 3: text summarization evaluation at NTCIR Workshop 4. In N. Kando & H. Ishikawa (Eds.), *Proceedings of the Fourth NTCIR Workshop on Research in Information Access Technologies: Information Retrieval, Question Answering, and Summarization*. Tokyo: National Institute of Informatics.

Li, W., Liu, J. Y., & Wang, C. (2005). Web document duplicate removal algorithm based on keyword sequences. In *Proceedings of 2005 IEEE International Conference on Natural Language Processing and Knowledge Engineering (IEEE NLP-KE '05)* Oct. 30 – Nov 1, 2005, Wuhan, China, *(*pp. 511–516). Piscataway, NJ: Institute of Electrical and Electronic Engineers.

Lopresti, D. P. (1999). Models and algorithms for duplicate document detection. In *ICDAR '99 Proceedings of the Fifth International Conference on Document Analysis and Recognitio*n (pp. 297-300). Washington, DC: IEEE Computer Society.

Minutillo, S. (2007). Welcome to Feed on Feeds, your server side, multi-user RSS and Atom aggregator! Retrieved from www.feedonfeeds.com on July 12, 2011.

Parman, R., & Sneddon, G. (2008). What is SimplePie?, Retrieved from simplepie.org/wiki/faq/what_is_simplepie on July 12, 2011

Rollett, A., & Wood, A. (2009). Rnews Feed Aggregator. Retrieved from rnews.sourceforge.net on July 12, 2011.

Universal McCann. (2008). *Power to the people, social media tracker wave 3.0*. [Report]. New York: Universal McCann, pp.12–13.

Svore, K. M., Vanderwende, L., & Burges, C. J. (2007). Enhancing single-document summarization by combining RankNet and third-party sources. In *EMNLP-CoNLL 2007: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, June 28–30 2007, Prague, Czech Republic (pp. 448–457). Stroudsburg, PA: Association of Computational Linguistics.

Takeda, T., & Takasu, A. (2008). News aggregating system with automatic summarization based on local multiple alignment. In *Proceedings of the 6th International Conference on Informatics and Systems* (pp. 65–73). Cairo: Faculty of Computers & Information, Cairo University.

Wang, D. Z., & Chen, Y. H. (2007). Near-replicas of web pages detection efficient algorithm based on single MD5 fingerprint. In A. Aggarwal (Ed.), *ICAI'07*: *Proceedings of the 8th Conference on 8th WSEAS International Conference on Automation and Information*, (Vol. 8), (pp. 318–320). Stevens Point, WI: World Scientific and Engineering Academy and Society (WSEAS).

Wang, M. Y., & Liu, D. S. (2009). The research of web page ee-duplication based on web pages reshipment statement. In J. Chen & Z. Hu (Eds.), *Proceedings of the First International Workshop on Database Technology and Applications DBTA 2009*, 25–26 April 2009 (pp. 271–274). Piscataway, NJ: IEEE Computer Society.