**A General Simhash-based News Aggregator System**

**Pengcheng Hu1,a,\*, Xiangdong You1,b**

1Institute of Problem Solving, XYZ University, My Street, MyTown, MyCountry

a hupengcheng1993@126.com, b email

\*corresponding author

**Keywords:**The News Aggregator, News Service, Simhash

**Abstract.** News aggregator is a system which aggregates online newspapers, blogs, videos from different sources, and recommend news to different users according to their interests. The system will crawl ten thousands of news from Internet every day, and try to recommend certain or similar news for users in a database which contains millions of news. We proposed a general simhash-based aggregator system to handle common problem like depulication, similar news retrieval and big event detection.

1. **Introduction (Heading 1)**

An emerging trend in modern press industry is providing digit news to readers. Users can browse news on a website belongs to a certain news service provider, however, the website just show the same contents to all users regardless of their interests and browsing history. Meanwhile, users need to browse different websites if he subscribes several news services.

A news aggregator is a system which aggregates online newspapers from different sources and provide personalized recommendation to users. The system need to crawl ten thousands of news daily from different news sources and clean them into well-formatted text. Then the system also need to calculate similarity between any two news so that it can recommend similar news to users according to their browsing history. A news aggregator can also provide special reports which contains series of news of a continuous event from different news provider. At the same time, news aggregator is an automatic system of news recommendation, it should be capable of detecting big events like natural disasters or political scandals when no editors involved in.

Deduplication is crucial when news aggregator leverages web crawler to collect online news. Web crawler, also called spider, is an Internet robot that systematically browses the Internet to collect information. It usually starts with a list of URLs to visit and crawl all contents of these website pages. It will identify all hyperlinks in the page and adds them to the list of URLs to visit, and repeat the process until there is no URLs in the list. There may be multiple pages contain the same hyperlink which link to the same webpage and it’s wasteful to visit the same URLs several times. It’s easy for us to handle this problem by using a cache table like a Bloom filter, however, it’s hard for us to tell whether two different websites have highly similar contents, and we called this near-duplicates. It’s common that one news provider will reproduce the same news from another provider. So it’s essential for news aggregator system to identify near-duplicative news.

It’s also essential to recommend similar or related news to users when they have finished browsing a piece of news. An implicit assumption of user’s browsing behaviour is that users tend to read news of similar topics. It’s reasonable to recommend similar news to users according to the news he is browsing. Content similarity of a set of news text is key to this kind of recommendation.

News aggregator relies heavily on automatic algorithms and it’s crucial for aggregator systems to be capable of detecting unusual big events in time when there is no human editor involved in. The most import part of big event news is timeliness and we can’t emphasis it too much. News aggregator should also provide special reports of continuous event which can be collected from different news provider.

Most aggregator system will provide category label for each news like Politics, Entertainment and Sports, etc. The system usually mapping text of each news into a fix-length vector and training a multi-class classifier which use these vector as features. Similarity can also be calculated by cosine distance of two feature vectors. However, a big challenge of news aggregator system is the issue of scale: the system will index billions of news in the database and crawl thousands of news per day. In this thesis, we proposed a general framework to handle deduplication, similarity searching, big event detecting and special reports generation very quickly in billions of news.

Feature vector representation of news

The news aggregator system need to map the text of each piece of news into a fix-length vector before further processing. There are many different manners for mapping, however, BoW models and distributed representation is the most common methods used in NLP. TF-IDF is one of the most popular representation of BoW models, it’s simplistic but surprisingly useful in practice. Another dominated method is distributed representation which map text into highly compressed vectors and doc2vec is the most popular representation.

Preprocessing is necessary regardless of which representation the system will adopt, TF-IDF or doc2vec. Preprocessing usually contains several sequential steps including tokenization, remove stop words and stemming. Tokenization is various for different languages and particularly difficult for languages written in scriptio continua which exhibit no word boundaries such as Ancient Greekm and Chinese. Stop words means that the frequency of these words is too high and we can benefit little from these words, so we can just remove these words from the text. Stemming is the process of reducing derived words to their word stem, base or root form. For grammatical reasons, two semantic similar sentences may use different forms of word, it would be useful to turn these words into a common base form.

We also need to reduce feature dimension after extracting feature representation, because high dimension feature will do damage to performance. Dimensionality reduction is the process of reducing the number of feature vector’s dimension. It can be divided into feature selection and feature extraction. Feature selection approaches try to find a subset of the original features according to the weights of each feature. However, feature extraction usually tends to keep each feature by mapping the feature vector from a high-dimensional space to a low-dimensional space. PCA (principal component analysis) and LDA (linear discriminant analysis) are common transformation techniques.

SimHash value calculated

The news aggregator system will map text of each piece of news into a fix-length feature vector. The similarity between two news can be calculated by cosine distance of their vectors, however, it’s hard for us to find the top N news with highest similarity when specific news provided. The time complexity of similar news retrieval grows rapidly when the size of the indexed news continuously increases. We can calculate the similarity between each news with the current news, but the time cost is too huge to accept. By leveraging simhash, we can find most similar news in constant time in a database which contains billions of news.

Simhash was proposed by Charikar in[], it’s a dimension reduction technique which can map high-dimensional vectors to small-sized fingerprints. In section 2, we have discussed how to extract feature vectors for each piece of news and these vectors usually have high dimension more than 30k. We can use simhash to map these vectors to a f-bit fingerprint where f is small, say 64 or 128. One import property of simhash is: Similar feature vectors will have similar fingerprints. Similar fingerprints means that the Hamming distance of these two fingerprints is small, say d. So we can retrieve similar news using news’ simhash fingerprints.

Suppose that the dimension of feature vector extracted from news text is D, and we used the same implementation like Charikar to generate f-bit fingerprint as follows:

We initialize an array V of length f , each element of V is set to 0. For each feature in the feature vector, we use a hash function to hash the feature into a f-bit hash value. We use the f-bit hash value to update the array V: if the i-th bit of the hash value is 1, we increase the i-th element of V by 1; if the i-th bit of the hash value is 0, we decrease the i-th element of V by 1. After we processed all features in the feature vector, values of elements in V are positive or negative. We generate the f-bit fingerprint F according to the sign of elements in V: set the i-th bit of F to 1 if the i-th element of V is positive, or the i-th bit of F will be set to 0.

Now that we have generated a f-bit fingerprint for each feature vector, how do we quickly retrieve other fingerprints that are similar with the given one? Similar fingerprints generated by simhash means that the Hamming distance of these fingerprints are small, says d. How do we quickly retrieve fingerprints that differ in d bit-positions?

Suppose we have 10 millions of news stored in the aggregator system and the fingerprints is 64-bit. The number of fingerprints is closed to 223 if the system do simhash on each piece of news. We can also assume that the Hamming distance of two fingerprints, F and F’, should be small than a threshold, says d, if two news are semantic similar. We can conclude that |f-d| bit-positions are same for F and F’, although we have no idea the exact d-positions where two fingerprints differs. We can separate the f-bits into N blocks, where N > d, then there must be at least one block that all bits are same for F and F’.

By leveraging this finding, we can maintain many tables and each table is associated with one block of f-bit fingerprints. Two fingerprints will be stored in the same table if they have identical block. We can retrieve similar fingerprints just in N tables rather than all tables, then what matters it the size of each table. Let’s consider the following 3 different designs:

1. Split 64 bits into 8 blocks, each contains 8 bits. If we have 223 fingerprints, then there will be 223-8 = 32768 fingerprints in each table. On average, a probe retrieves 8 \* 32768 = 262144 fingerprints.
2. Split 64 bits into 4 blocks, each contains 16 bits. There will be 223-16 = 128 fingerprints in each table. On average, a probe retrieves 4 \* 128 = 512 fingerprints.
3. Splits 64 bits into 5 blocks having 13, 13, 13, 13 and 12 bits respectively. Each table will have 223-13 = 1024 fingerprints and a probe will retrieves 5120 fingerprints on average.

We carried out two experiments for design 2 and 3, because the probe space of these two design choices are acceptable. In our experiments, we used Blob Service of Microsoft Azure which is a NonSQL database. The Blob Service contains lots of blobs and each blob is identified by a key. A blob is just like a table, it can hold any objects and each object should also have a key.

Our aggregator system have 9.6 millions of news and retrieve

Deduplication, Similar news retrieval and Big event detection

We have discussed generation and retrieval of simhash fingerprints, then how do we use this technique to handle deduplication, similar news retrieval and big event detection?

Manku states in [] that Hamming distance of near-duplicates is close to 3. And we also found that two news are similar rather than near-duplicative if the Hamming distance is bigger than 3 but small than 8.

For deduplication, we will retrieve similar fingerprints whenever we crawled a new piece of news. If there are fingerprints that differs only in 3 or less bit-positions with the new crawled news, we will not persistent the new crawled news into our database.

And when we want to recommend similar news to readers, we will retrieve similar fingerprints and only keep fingerprints if Hamming distance is small than 8 and bigger than 3. Then we will recommend corresponding news to readers, and we can rank these news according to the similarity with the currently reading one.

Big events, like natural disasters or political scandals, usually happened unexpectedly. Big events will be paid great attention and the public want to know and discuss all details of the event. Different news providers will report the event and huge amount of news will be reported in a short time. If the aggregator system found that lots of similar news was crawled in last few hours, we can assume that the system has detected a big event. So the system can collect all these related news into a special report and recommend it to all readers.

Conclusion

In this thesis, we proposed a general news aggregator framework which leverage simhash to handle deduplication, similar news retrieval and big event detection. By calculate simhash fingerprints on the feature vector of news rather than original text, any aggregator can integrate these functionalities into it’s running system without extra efforts to alter existing framework.

* + 1. **Hamming Distance and Similarity**

Our aggregator system handles deduplication and news retrieval by Hamming distance of two fingerprints. We randomly sample 10000 news from our database and retrieve fingerprints which has k different bits with the given one, where k differs from 1 to 10. For fingerprint F and F’ which is stored in the same blob as F, we calculated cosine distance of original feature vectors that F and F’ was generated from, if Hamming distance is k. Figure 1 plots relationship of similarity and Hamming distance.

1. **Manuscript Preparation (Heading 2)**

We strongly encourage authors to use this document for the preparation of the camera-ready. Please follow the instructions closely in order to make the volume look as uniform as possible.

Please remember that all the papers must be in English and without orthographic errors.

Do not add any text to the headers (do not set running heads) and footers, not even page numbers, because text will be added electronically.

For a best viewing experience the used font must be Times New Roman, on a Macintosh use the font named times, except on special occasions, such as program code.

* 1. **Page Setup (Sub-Heading 2.1)**

The paper size must be set to A4 (210x297 mm). The document margins must be the following:

* Top: 3.5 cm;
* Bottom: 2.5 cm;
* Left: 2 cm;
* Right: 2 cm.

It is advisable to keep all the given values.

Regarding the page layout, authors should set the Section Start to Continuous with the vertical alignment to the top and the following header and footer:

* Header: 1.5 cm;
* Footer: 1.5 cm.

Any text or material outside the aforementioned margins will not be printed.

* 1. **First Section (Sub-Heading 2.2)**

This section must be in one column.

* + 1. **Paper Title**

Use 20-point type for the title, aligned to the center, linespace exactly at 14-point with a bold and italic font style and initial letters capitalized. No formulas or special characters of any form or language are allowed in the title.

Words like “is”, “or”, “then”, etc. should not be capitalized unless they are the first word of the title.

* + 1. **Subtitle**

Use 16-point type for the subtitle, aligned to the center, linespace exactly at 14-point with bold and italic font style.

Words like “is”, “or”, “then”, etc. should not be capitalized unless they are the first word of the subtitle. The initial letters should be capitalized. No formulas or special characters of any form or language are allowed in the subtitle.

* + 1. **Authors**

Author(s) name(s) should be aligned to the center with linespace single. The text must be set to 12-point and the font style set to bold.

There should be a spacing before of 6-point.

* + 1. **Affiliations**

Affiliation(s) should appear aligned to the center including organisation, address and e-mail.

The linespace must be single with 12-point of font size and the font style set to italic.

* + 1. **Keywords**

Each paper must have at least one keyword. If more than one is specified, please use a comma as a separator. Keywords should appear justified, with a linespace single, spacing before of 18-point, spacing after of 18-point and font size of 12-point.

The sentence must end with a period.

* + 1. **Abstract**

Each paper must have an abstract. The abstract should appear justified, with a linespace single, spacing before of 18-point and after of 60-point, and font size of 12-point.

The sentence must end with a period.

* 1. **Second Section**

This section must be in one column.

The section text must be set to 12-point, justified and linespace single.

Section, subsection and sub subsection first paragraph should not have the first line indent, other paragraphs should have a first line indent of 0,5-centimeter.

* + 1. **Section Titles**

The heading of a section title must be 12-point bold, aligned to the left with a linespace single and an additional spacing of 10-point before and 10-point after. The initial letters should be capitalized.

Dot should be included after the section title number.

* + 1. **Subsection Titles**

The heading of a subsection title must be 12-point bold with initial letters capitalized, aligned to the left with a linespace single and an additional spacing of 10-point before and 10-point after.

Words like “is”, “or”, “then”, etc should not be capitalized unless it is the first word of the subsection title.

Dot should be included after the subsection title number.

* + 1. **Sub Subsection Titles**

The heading of a sub subsection title should be in 12-point bold with initial letters capitalized, aligned to the left with a linespace single and an additional spacing of 10-point before and 10-point after.

Words like “is”, “or”, “then”, etc should not be capitalized unless it is the first word of the sub subsection title.

Dot should be included after the sub subsection title number.

* + 1. **Tables**

Tables must appear inside the designated margins.

It is advisable the use of text boxes in this case.

Tables should be properly numbered, centered and should always have a caption positioned above it. Captions with one line should be centered and if it has more than one line should be set to justified. The font size to use is 9-point. No bold or italic font style should be used. Spacing before and after should be of 12-point and 6-point, respectively.

The final sentence of a caption must end with a period.

Table 1 This caption has one line so it is centered.

|  |  |
| --- | --- |
| Example column 1 | Example column 2 |
| Example text 1 | Example text 2 |

Please note that the word “Table” is spelled out.

* + 1. **Figures**

Please produce your figures electronically, and integrate them into your document.

Check that in line drawings, lines are not interrupted and have a constant width. Grids and details within the figures must be clearly readable and may not be written one on top of the other.

Figure resolution should be at least 300 dpi.

Figures must appear inside the designated margins.

It is advisable the use of text boxes in this case.

Please do not use indentation and set the figure layout to in line with text.

Figures should be properly numbered, centered and should always have a caption positioned under it. Captions should be centered. The font size to use is 12-point. No bold or italic font style should be used. Spacing before and after should be of 6-point and 12-point, respectively.

The final sentence of a caption must end with a period.



Figure 1 This caption has one line so it is centered.



Figure 2 This caption has more than one line so it has to be set to justify.

Please note that the word “Figure” is spelled out.

* + 1. **Equations**

Equations should be placed on a separate line, numbered and aligned to the right. An extra line space should be added above and below the equation.

The numbers accorded to equations must appear in consecutive order inside each section or within the contribution, with number enclosed in brackets and set on the right margin, starting with the number 1.

The use of a table with two columns is advisable.

Example:

a = b + c (1)

* + 1. **Program Code**

Program listing or program commands in text should be set in typewriter form such as Courier New.

Example of a Computer Program in Pascal:

Begin

Writeln('Hello World!!');

End.

The text must be aligned to the left with the linespace set to single and in 12-point type.

* + 1. **Reference Text and Citations**

All references should be numbered in square brackets in the text and listed in the References section in the order they appear in the text.

References should be set to 10-point, justified, with a single linespace and hanging indent of 0,5-centimenter.

1. **Conclusions (Heading 3)**

We hope you find the information in this template useful in the preparation of your submission.

**Acknowledgements**

If any, should be placed before the references section without numbering.

**References**

[1] Maganioti, A.E., Chrissanthi, H.D., Charalabos, P.C., Andreas, R.D., George, P.N. and Christos, C.N. (2010) Cointegration of Event-Related Potential (ERP) Signals in Experiments with Different Electromagnetic Field (EMF) Conditions. Health, 2, 400-406.

[2] Bootorabi, F., Haapasalo, J., Smith, E., Haapasalo, H. and Parkkila, S. (2011) Carbonic Anhydrase VII—A Potential Prognostic Marker in Gliomas. Health, 3, 6-12.

**Appendix**

If any, the appendix should appear directly after the references without numbering, and not on a new page.