International Journal of Computer (IJC)
ISSN 2307-4523 (Print & Online)

http://ijcjournal.org/

Aggregating News Articles – NewsInn Algorithm

Radu Nicoara*

NewsInn, Bucharest, Romania
Email: nicoaraeradu@yahoo.com

Abstract

NewsInn is an algorithm designed to aggregate articles from multiple news sources into one general, story-based meta-article. This entity is then displayed by calculating the most relevant news article in relation to it, and by listing underneath it the remainder of articles. Using a simple User Interface, the algorithm makes it easy for end users to understand current events, the coverage that they have and the impact that the story carries.

Using multiple Natural Language Processing techniques, Newsinn manages to parse 19 different news source and aggregate more than 700 articles daily into their respective meta-articles. Seeing how even a human would have issues deciding if certain articles are related or not, as it depends on what criteria is used for making this decision, NewsInn uses different parameters to control the level of aggregation.

Keywords: News Aggregation; Text Classification; Natural Language Processing; Artificial Intelligence.

1. Introduction

In order to exemplify the need for a news aggregation algorithm, I will take the case of a stock trader. He needs to go everyday through a multiple news sources in order to make sure he does not miss a potentially important information. The issue however is that every time he has to read about the same events written from different points of view. A single app that would provide event-based information, and then would allow the user to research further only into the story that he is interested in would prove to be efficient to such users. I chose to use the following sources as a base for the algorithm, taking into account the reader reach of the website and the relevance for the news section: *BBC, Reuters, New York Times, Washington Post, The Guardian, Al Jazeera, Wall Street Journal, The Week, CBS News, ABC News, LA Times, The Telegraph, Time, Russia Today, Fox News, Euro News, NBC, Global News* and *Euro News.*

---

* Corresponding author.

The first step of the algorithm is to use a crawler for parsing the sources [5]. The article has its HTML tags stripped and the raw text is saved into the database. Together with this, I save the title and position of the article inside the news page. As to be expected, articles that are placed higher up in the page have more relevance than the ones lost in the bottom links.

I used an adapted version of *Sentiword* [2] to extract the opinion in the articles. This library provides a score from 0 to 1 for the positive and negative connotations of each word. In addition to removing a big part of the library, I needed to add words that were usually found in the articles, words related to terrorist attacks or economic fluctuations for example. Since the title was chosen by a human being, which are way better than machines at identifying sentiments, the score of the title had a bigger influence on the total score [7].

After multiple tests, the final score was:

$$TotalOpinion = 100 * \frac{\left(\sum AO_{poz} + 2.2 * \sum TO_{poz}\right) - \left(\sum AO_{neg} + 2.2 * \sum TO_{neg}\right)}{Nr.Words}$$

where *AO* = Article Text Opinion, *T O*= Title Opinion, *Nr. Words* = number of words inside the article

Using the same principle, I use the algorithm to classify the articles into Sport, Economic, Politic or General category news [6]. As can be observed in the official website [2], the algorithm rarely miss qualifies an article.

2. Extracting KeyWords

In order for the algorithm to understand the main event that the article portrays, it first needs to extract a number of keywords. As a result, a text like „*The president X of Y country began the process of integrating Z project into the public agenda*" would yield *X, Y, Z* and „*integrate*" as keywords results. This is actually similar to the way a human would perceive the article as well [8,9]

I used *OpenNLP* [4] to perform all the needed operations on the text. The first one is the extraction of named entities, such as corporations or agencies. The second process is the extraction of the names of persons. Afterwards, we extract the geographic locations from the article. We then count the number of times the extracted keywords appears in the text. Since they are more likely to be relevant, all of them receive a higher count weight.

The following step is stemming all the words in the text. Counting the occurrences of all phrases provides a good representation of the important used keywords in the article. In order for the processing to take as little time as possible, we trim the text to a maximum length of 25,000 words.

I use a synonym dictionary to aggregate different similar keywords into a single entity, and then use an acronym library developed specially for this project to map different short-named entities into a single keyword.

Out of all the resulted keywords, we save only the 13 most relevant ones, based on the score described below.

We remove prepositions and articles, normalize abbreviations and generally ignore common words, as they have the tendency to show up in any type of article, and we are looking for unique identifiers.

A necessary step for the end user is article summarization [7]. Using the keywords extracted earlier, we calculate which two parses in the text have the most relevance both in relationship with the keyword array and within itself. I then, using OpenNLP [4] calculate the text-wide occurrence for each word in each sentence. The sentences with the highest score are then cosmeticize and shown as the article's summary [5].

Figure 1: The result of keyword extraction, sentiment analysis and article summarization, as seen in UI

The final score for each keyword is the score shown below:

$$KeywordScore = entityScore * NumOcc * titleScore * \log(\ln(PastOcc + 3) + 10)$$

Where $entityScore$ is a constant raging from 0.9 to 1.5, $NumOcc$ is the number of instances in which the keyword appears in the text, $titleScore$ is 3 if the keyword appears in the title of the article and $PastOcc$ is the number of times the keyword has been extracted in the past week

3. Aggregating The Articles

In the following lines I define a meta-article as being a conglomeration of one or more news pieces. It is the final entity that will be displayed to the end user. It is defined by a number of five keywords, and the most relevant article is displayed as being the image, title and summary of the whole story.

Having all the data extracted in the chapter above, the algorithm takes the first 6 most relevant keywords and create a proto-meta-article out of it. It then parses all articles and sees which entity it would best fit into, with the condition of having at least 3 of the keywords in common. Since this algorithm is likely to favorise articles that are parsed first, I run a threaded process multiple times to avoid this bias.

Because there are a large number of parameters which can adjust the aggregation process, I choose the one setup which would lead to the smallest number of miss-conglomerations occurrences. The downside of this is that it sometimes leads to meta-articles being split up based on different points of view on the subject, or most often, different contexts.

After the conglomeration is complete, we calculate the average opinion for each meta-article, together with the score for the sport, politics and economy section. We create a meta-score based on the following:

- How many articles are in each meta-article
- How relevant is each article to the meta-article keywords
- How recent the newest aggregated articles are
- How on point the articles are within themselves. To determine this, we use a score taken from the summarization algorithm

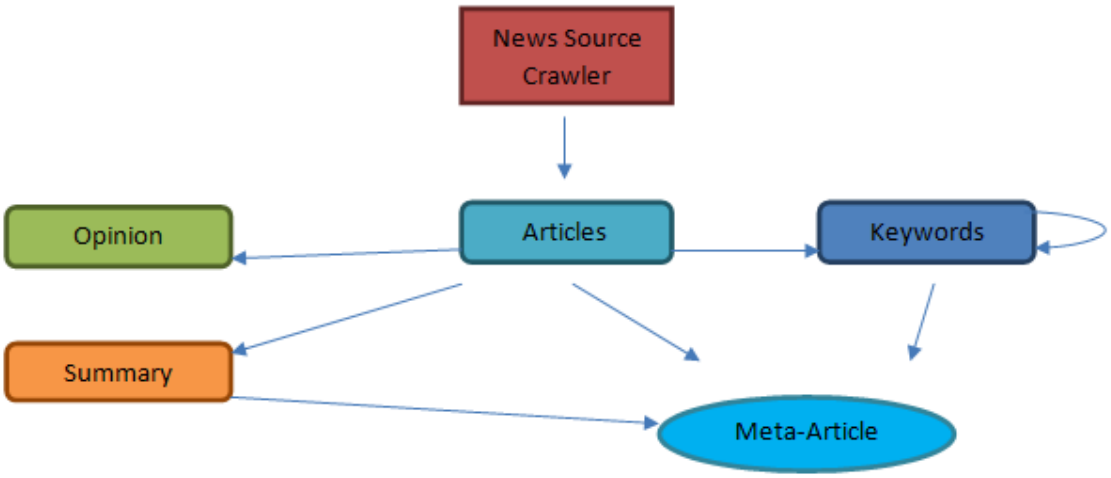

Figure 2: A visual representation of the main processes behind the NewsInn algorithm

4. Results

As can be seen in the official webpage of the NewsInn project, the algorithm manages to conglomerate the articles without difficulty, leading to only a small number of false positive aggregations. The downside of this, however, is that sometimes one story gets put into 2 or 3 meta-articles, instead of one. The explanation for this behavior is that the algorithm sees multiple points of view about the event.

Below I will show a statistic of the data processed and displayed on the 13th of July 2016

- Total number of articles on display: **2540**

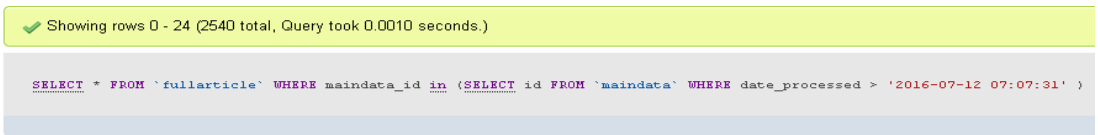

Figure 2: The number of articles displayed on the extraction date

Next I went by hand through all the meta-articles and the linked news articles and came up with the following data regarding the aggregation of news pieces:

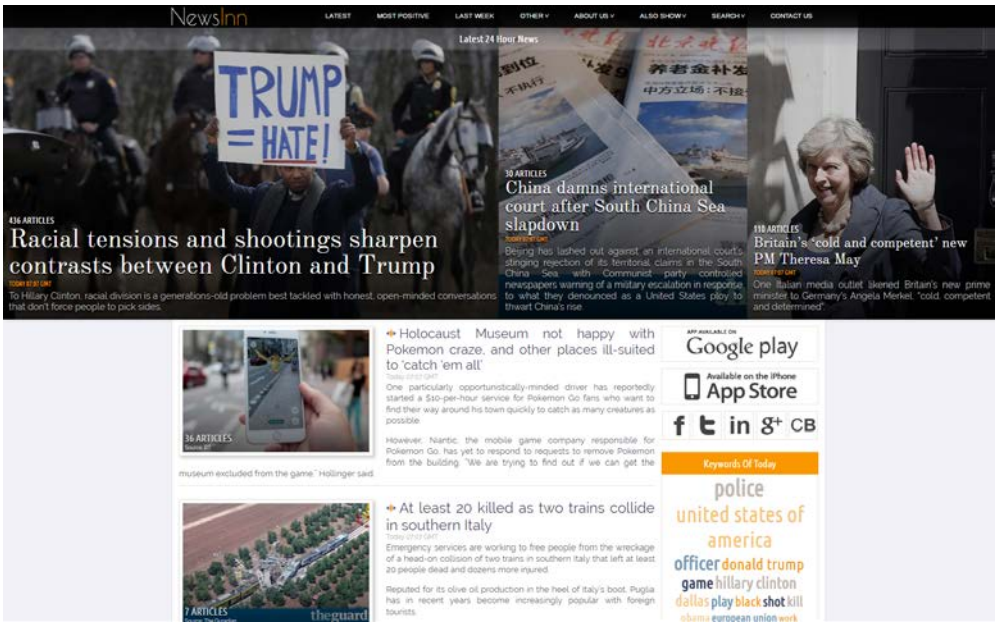

Figure 3: The user interface of the article, as seen on the official web page [2]

- The number of falsely aggregated articles (false positive, or articles that tell a different story than the one conglomerated) was **43 (1.69%).** This result clearly proves that the algorithm is very good at doing what it was designed to, which is to minimize false positive conglomeration. The down side of choice will be clearly revealed in the following section.
- Articles that were supposed to be conglomerated into an existing meta-article, but were either put into a separate meta-article, or left un-aggregated: **145 (5.70%)**. This may not seem like much at first glance, but it is actually **18.8%** of the number of **768** articles processed in the latest 24 hour period. The explanation is that not all stories live only 24 hours, so the relevant articles tend to accumulate over a period of several days.
- Total number of meta-articles created was **697**
- Number of meta-articles with only one piece extracted was **612**

Modifying the script so that it is tending to conglomerate more data leads to an increased amount of false positives, which in turn would cause a lessening of the perceived value by the end user.

5. Conclusions

The results calculated in the previous section shows both the current capabilities of the algorithm, and the extent of improvement that is still needed. The essential part of the algorithm is show to work quite well, however, there is still room for advances in the aggregation cycle.

Given the amount of data being processed, any improvement in the algorithm would need to have a fundamental

principle at the basis. It would be tremendously difficult to use a trial and error approach. One such example is the comparison of each keyword against a synonym library, as this has led to a smaller pool of possible keywords without giving up on the false positive constraint.

Based on observations, it is very possible that NewsInn Algorithm will further be modified, either in order to suit changes in data structure, or to improve the aggregation capabilities.

References

[1] Radu Nicoara , *Algoritmi de Agregare si Procesare a Stirilor*, RO, 2016

[2] Esuli, Andrea, and Fabrizio Sebastiani. *Sentiwordnet: A publicly available lexical resource for opinion mining*. Proceedings of LREC. Vol. 6. 2006

[3] Baldridge, J. ,*The Apache OpenNLP project* , 2005

[4] Mani, Inderjeet, *Automatic summarization*, Vol. 3, John Benjamins Publishing, 2001

[5] Castillo, Carlos. *Effective web crawling.* ACM SIGIR Forum, vol. 39, no. 1, pg. 55-56, 2005.

[6] Ku, Lun-Wei, Yu-Ting Liang, and Hsin-Hsi Chen, *Opinion Extraction, Summarization and Tracking in News and Blog Corpora.* AAAI Spring Symposium, vol. 100107, 2006

[7] Hu, Minqing, and Bing Liu. *Opinion extraction and summarization on the web*, AAAI, vol. 7, pg, 1621-1624, 2006

[8] Brill, Eric, *Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging*, Computational linguistics 21, no. 4, pg 543-565, 1995

[9] Manning, Christopher D., and Hinrich Schütze. *Foundations of statistical natural language processing*. Vol. 999. Cambridge: MIT press, 1999