Coding the News: The Role of Computer Code in Filtering and Distributing News

Matthew S. Weber

Allie Kosterich

Department of Communication, Rutgers University, New Brusnwick, NJ USA 08901 matthew.weber@rutgers.edu

Abstract

This article examines the role of code in the process of news distribution and the degree to which code and algorithms can filter and prioritize news, much as an editor would. This article reports the results of an investigation of code contained in 59 open source mobile news apps and an analysis of the content of that code. Findings highlight the journalistic decisions made in code and contribute to discussion surrounding the relationship between algorithmic and traditional news values.

CCS Concepts

Theory of computation: Theory and algorithms for application domains

Keywords

Algorithms, News Curation, Algorithmic Curation, Journalism, Organizational Implications

1 Introduction

Prior research on computer code, algorithms and news production has focused on the producers of code [1], but the code itself is often imbued with agency to produce news [4]. The present exploration of code helps unlock both the algorithmic black box [3]. This manuscript highlights the role of algorithmic judgment – often in place of editorial decision-making – with regards to the curation of content and where communicative aspects are explicit and visible in plain sight (plain code). The discussion moves away from the common notion that code is replacing humans as producers of news and shifts toward the role of code in helping journalists order and communicate the news. The core research question addressed is the degree to which algorithmic values embedded in code decide how news is filtered for distribution. To address our core question, this article reports on an investigation of the code and algorithms contained in mobile news apps. Content analysis was used to study the computer code, and to identify key elements of the computer code responsible for filtering and sorting news.

2 Algorithmic Journalism

Social science research on algorithms has focused on effects of use, most commonly directed toward consumers and society at large, framing the public as in some way held to the power of the organization or algorithm implementer. To that end, Diakopolous (2013) questions consequences of the algorithm's inherent lack of transparency and potential bias toward consumers; whereas Gillespie [7] takes issue with the lack of any official interrogation of the role of algorithms in human information practices. Work recent years has shifted the focus to consider broader challenges associated with automatic curation of content, including studies exploring the process of interacting with algorithms [6], the use of algorithms to identify local news not covered elsewhere as well as the use of algorithms to curate the news that reaches core audiences [11]. A common theme is the notion that while algorithms are technological in nature, they are developed and recalibrated within complex social systems [5], making it difficult to thoroughly examine them as a technology with agency that affects human behavior or from the perspective of humans with agency utilizing a technology. Algorithmic journalism is particularly important in the context of mobile news applications. Mobile news apps have become a critical growth area for news organizations in recent years. Current data indicate that mobile applications are increasingly a central distribution channel for news consumption. To that end, recent reports show that mobile news applications have experienced tremendous growth. Exact tracking data is difficult to locate, however aggregated estimates from Yahoo!

show that mobiles news app traffic great 135% from 2014 and 2015, compared to the overall app traffic growth of 58% in the same period [9]. Moreover, every indication is the trend will continue; 2014 data shows that 90% of time spent online is spent on mobile news apps [8]. In sum, newspaper companies must tackle the challenges associated with building mobile news apps.

Although code can be designed to perform journalistic tasks, analysis has shown that there are still distinctions in the nature of the output. When code is used to curate content, for instance, challenges arise with regards to how articles are selected and what portion of the content is then presented to the user [12]. Nevertheless, the technical tasks can be mimicked and replicated by natural language–oriented computer programs. Moreover, code is increasingly becoming a mediator of news [2], underscoring the need to interrogate and examine how code is a tool being integrated into traditional organizational functions. The code in mobile news apps takes in the work produced by journalists and sorts and ranks that content based on decisions made by the programmer. The following sections delve into an examination of how computer code and algorithms mediate the journalistic process.

4 Method and Analysis

To identify appropriate apps for analysis, the researchers cast a wide net in selecting the mobile news apps. Keyword searches were conducted across platforms (google.com, bing.com, yahoo.com and github.com). GitHub was included because it is a major code repository, although search functions are more limited on the site. Key search terms were: news, journalism, mobile app, mobile application, open source.

Following the methodology of traditional content analysis, this study translated this methodology and applied it in the context of computer code. The two coders were both fluent in the core languages (Java, SQL, Python, Objective C) used for the mobile apps under analysis. Coding was conducted with a series of shared Excel spreadsheets, as well as GitHub as repository for viewing and extracting code. A critical reading was undertaken, whereby the authors sought to connect key collections of the data to larger issue-relevant themes. A categorical aggregation technique was used to assess the issue-relevant meanings that exist between the data and the research framework. Coders reviewed more than 3,000 lines of code and identified core functions that performed a journalistic process (sorting, filtering or prioritizing the presentation of news articles). Both coders first coded five of the 59 applications; the results were then reviewed and any discrepancies in identifying journalistic functions were discussed.

5 Results

Code was collected for 59 mobile applications and 134 instances were identified where code performed journalistic functions. Of the 59 apps analyzed, 83% are applications that are available for download as ready to use mobile apps. The other 17% are apps that are either demonstrations projects or open source libraries that can be compiled to function as an app. These is a relatively high degree of variance in the apps that are available; the number of estimated annual downloads (based on Android App Store estimates estimates) ranged from 100 to 500,000 (M = 36,158, SD = 62,633). The majority of applications used Python as the primary programming language (58%). The second most prominent programming language was Java (31%). The remaining applications were programmed in Objective C or C++ (9%).

5.1 Primary Action Categories

The initial review of the code contained in the mobile apps analyzed for this research revealed a set of primary action categories that were performed by the majority of mobile news applications. These categories are initial information gathering, pre-processing, training, algorithm, and output. The *information gathering* stage occurs when the code within the application asks the user for data based on user interests or through access to social media profiles. Once initial data are collected, a second set of code is activated that *pre-processes* the input data to present the user with initial articles that may be of interest. In the *training* phase, those initial articles are presented to the user as a "training model." Feedback is obtained and stored in the application. The initial input data are now disregarded and the application focuses on the feedback obtained from the user. The user feedback is used to set initial conditions for the *algorithm*. of the algorithms are a variant of machine learning models, which means that the algorithms take the input as a starting point and use a set of rules to clarify new news articles and to serve those articles out to the user (*output*). The user will

continue to interact with the output over time, reading some articles and ignoring others. The following sections present details of the specific results pertaining to each step of the algorithmic process.

5.2 Initial Information

The first information that is collected is used to initiate the data collection phase of the process. Often this data collection takes the form of collecting information from social media. In the apps reviewed, 43% relied on social media outlets such as Twitter, Facebook and Reddit for preliminary information to help provide initial conditions for the provision of news. In the initial step, the user is queried for some kind of initial input. With the initial user input in hand, the application then turns to other sources to retrieve initial news and information. Data collection sources prominent in applications analyzed include sources such as The New York Times, Hacker News, Huffington Post and Google News. For example, the following snippet of code, from The Guardian's Secure Reader Project, and written in the Scala programming language, provides a number of different pre-programmed options to retrieve articles from specific sources. For instance, the following delineates a category that allows the user to select to gather news from Voice of America's news service:

Algorithm Sample 1: Command to set access to the Voice of America API

```
<outline category="en_US" text="VOA News" htmlUrl="http://www.voanews.com/api/epiqq"
xmlUrl="http://www.voanews.com/api/epiqq" />
```

The code here is relatively simple, in that it specifies this news is for the "en_US" or US English category, and provides the API that would be used to retrieve news for Voice of America. This type of retrieval is common and takes in an initial set of news from a third-party site such as Voice of America; when the API is used, it also retrieves the date of stories and section the stories appear in.

5.3 Pre-processing and Training

Data collected from third-party sites, including news articles from sites such as The New York Times, are retrieved in a format often determined by the third party. The next step is generally to clean the data to store it in a format easier for the app to manage. This process includes eliminating stop words, cleaning data of problematic characters, and formatting the data. The elimination of stop words refers to the removal of common language terms not helpful for machine learning. For instance, words such as "as" "is" and "the" do not provide useful information for automation, and thus are often eliminated. At this point in the process, the mobile application has gathered input from the user, potentially collected initial data from social media, collected an initial list of stories from pre-determined third-party news websites, and cleaned the data so that it is compatible with the mobile application. Then, the code produces a training set for the algorithms.

5.4 Algorithms

The algorithm is the set of coded rules that takes the training set and the news articles and pairs the two together to determine what news articles to return to consumers. For instance, the following code calls on a well-known algorithm, naïve bayes - specially the multinomial variant of naïve bayes, which considers a wider range of variation in the data.

In the first step, the algorithm works through training data, updating variables based on data fed from user interaction with a story (reading vs. not reading, stored in story_data).

Algorithm Sample 2: Storing of training data

```
 x_train = np.append(x_train,[story_data])
('clf', MultinomialNB())])
 classifier.fit(x_train, y_train)
```

The above example is drawn from a set of code that can be used in mobile apps to extract news from Hacker News and Reddit, and to customize the type of stories that are selected and how those stories are presented to the reader. Naïve bayes is a fairly simply group of classifiers that uses input data as training to determine the probability that new items will fall into particular categories; ongoing training is used to improve the accuracy of the probabilistic classification.

The last example uses a machine-learning algorithm referred to as k-nearest neighbors (KNN).

Algorithm Sample 3: Sample of KNN Classification

```
parameters = {'algorithm':('ball_tree', 'kd_tree', 'brute'), 'n_neighbors':[5, 50, 500]}
clf = grid_search.GridSearchCV(KNeighborsClassifier(), parameters)
clf.fit(features, label)
```

KNN is a another standard algorithm. The algorithm takes all of the input data and the key variables (e.g., likes, keywords) as input. In this case, the input data are labeled “features” and “label.” The training data are then mapped on the two specified dimensions (e.g., number of likes and number of keywords that match user preferences). The KNN algorithm then returns a cluster of stories that are relatively similar to one another and match closely to both dimensions. The numbers that are inputted into the algorithm (5, 50, 500) specify a tolerance for error.

5.5 Output

The final step is relatively straightforward. The mobile application outputs the content of the news articles to the mobile application interface. The application tracks the articles that a user selects and views, which provides a new input for the application and is used to refine the training set for the algorithm.

6. Discussion

The technical determination of output via news apps is generally framed through an input-throughput-output (I-T-O) model [10]. The basic concept is to take specific input, process it and modify it in some way, and then output it to the user. To this end, the prior section outlines the ways in which computer code and algorithms work together to collect initial information, collect user data, pre-process that data, train a system to produce content, generate new content based on a set algorithm, and then output the data to the user to continue learning and improving. Although many of the results rely on interpretation of computer code, the proceeding presentation of results is intended to provide the reader with a relatively digestible examination of the content of mobile news application code. In analyzing the content of the mobile news applications that were examined, a number of important themes emerge. The following discussion focuses specifically on the way in which computer code is imbued to perform a boundary spanning function with regard to the delivery of content, and the way that code is used to find news.

6.1 Code and Boundary Spanning

In performing boundary spanning work, transitioning between journalism and programming, code is imbued with agency to determine how news is sorted and distributed to consumers. For instance, in one of the above examples, the code specifies that user likes from Facebook will be a key filter for determining what news is presented to the user. In this way, code has the capacity to perform certain functions as determined by the programmer of the application. In previous contexts, such as the sorting of hard news at a daily news meeting, editors would meet to discuss the stories of the day and sort that content; the algorithm here is replacing that function with a matching of news article keywords and Facebook likes. To this end, the algorithms reviewed in this content analysis can be grouped into two categories: classification and regression, which are two main types of machine learning kernels (or classes). Classification and clustering (e.g. KNN) focuses on grouping the inputted content based on a set number of variables such as Twitter keywords and news article headlines. News articles are simply grouped and returned to the user. At a more sophisticated level, regression is used for quantifying the relevance of news articles. Basic examples such as SVM create a numeric score for news articles by using a regression to understand how closely news articles match a predetermined set of inputs. In each, the code and algorithm span the boundary between journalistic decision-making and computer programming. The code embodies a set of decisions that would be the domain of the journalist or editor were the code not in existence. It is

important to note that the technical domain of algorithms is vast, and the above is a gross simplification; for the purposes of this discussion, the above serves to illustration a key impact of algorithms in journalism.

7 Conclusion

In sum, this article provides the reader with significant insight into the technical process by which mobile news applications are designed. Ananny and Crawford [1] found that the emergence of programmers creating algorithms that determine the flow of news has brought about the emergence of a *liminal press*, whereby non-journalist actors are defining conditions under which news is produced and distributed. This article reveals a more nuanced reality through an examination of the content of mobile news app code. This work provides a glimpse into the nature of open source code, and provides a framework for future research connecting journalism and computer science.

References

[1] ANANNY, M. and CRAWFORD, K., 2015. A Liminal Press. *Digital Journalism 3*, 2 (2015/03/04), 192-208. DOI= http://dx.doi.org/10.1080/21670811.2014.922322.

[2] BURROWS, R.J., 2009. Afterword: Urban informatics and social ontology. In *Handbook of Reseach on Urban Informatics: The Practice and Promise of the Real-Time City*, M. FOTH Ed. Information Science Reference, Hershey New York.

[3] DIAKOPOULOS, N., 2014. Algorithmic accountability reporting: On the investigation of black boxes. *Tow Center for Digital Journalism, Columbia University*.

[4] DÖRR, K.N., 2016. Mapping the field of Algorithmic Journalism. *Digital Journalism 4*, 6 (2016/08/17), 700-722. DOI= http://dx.doi.org/10.1080/21670811.2015.1096748.

[5] ESLAMI, M., 2017. Understanding and Designing around Users' Interaction with Hidden Algorithms in Sociotechnical Systems. In *Proceedings of the Companion of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing* (Portland, Oregon, USA2017), ACM, 3024947, 57-60. DOI= http://dx.doi.org/10.1145/3022198.3024947.

[6] ESLAMI, M., ALEYASEN, A., KARAHALIOS, K., HAMILTON, K., and SANDVIG, C., 2015. FeedVis: A Path for Exploring News Feed Curation Algorithms. In *Proceedings of the Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative Work & Social Computing* (Vancouver, BC, Canada2015), ACM, 2702690, 65-68. DOI= http://dx.doi.org/10.1145/2685553.2702690.

[7] GILLESPIE, T., 2014. The relevance of algorithms. In *Media Technologies. Paths Forward in Socail Research*, P. BOCZKOWSKI and K. FOOT Eds. MIT Press, London, 167-194.

[8] KHALAF, S., 2015. Seven Years Into The Mobile Revolution: Content is King... Again. In *Flurry Analytics* Flurry by Yahoo!, Mountain View, CA.

[9] KHALAF, S., 2016. Media, Productivity & Emojis Give Mobile Another Stunning Growth Year. In *Flurry Analytics* Flurry by Yahoo!, Mountain View, CA.

[10] LATZER, M., HOLLNBUCHNER, K., JUST, N., and SAURWEIN, F., 2014. The economics of algorithmic selection on the Internet. *Media Change and Innovation Division Working Paper*.

[11] LEHMANN, J., CASTILLO, C., LALMAS, M., and ZUCKERMAN, E., 2013. *Transient News Crowds in Social Media*.

[12] LUSTIG, C., PINE, K., NARDI, B., IRANI, L., LEE, M.K., NAFUS, D., and SANDVIG, C., 2016. Algorithmic Authority: the Ethics, Politics, and Economics of Algorithms that Interpret, Decide, and Manage. In *Proceedings of the Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (Santa Clara, California, USA2016), ACM, 2886426, 1057-1062. DOI= http://dx.doi.org/10.1145/2851581.2886426.