The News that Matters to You
Design and Deployment of a Personalized News Service

Mark Stefik
Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, Ca. 94304
stefik@parc.com

Lance Good
Google, 6425 Penn Avenue, Suite 700, Pittsburgh, PA 15206
lance@goodle.org

Abstract

With the growth of online information, many people are challenged in finding and reading the information most important for their interests. From 2008-2010 we built an experimental personalized news system where readers can subscribe to organized channels of information that are curated by experts. AI technology was employed to radically reduce the work load of curators and to efficiently present information to readers. The system has gone through three implementation cycles and processed over 16 million news stories from about 12,000 RSS feeds on over 8000 topics organized by 160 curators for over 600 registered readers. This paper describes the approach, engineering and AI technology of the system.

Problem Description

It is hard to keep up on what matters. The limiting factor is not the amount of information available but our available attention (Simon 1971). In the context of news, traditional mainstream media coverage cannot address this issue. Although most people are interested in some of the topics covered in the mainstream, they also have specialized interests from their personal lives, professions, and hobbies that are not popular enough to be adequately covered there. A news service which can optimize our individual information foraging (Pirolli 2007) needs to be personalized to our interests.

A snapshot of a personal information appetite or "information diet" reveals further nuances. Some interests are enduring. Some interests are transient, following the events of life. When we form new interests, a topically-organized orientation helps to guide our understanding of a new subject area.

We believe that this approach will be of interest to information providers that want to grow their audience by providing personalized information delivery and targeting groups with focused interests. Specialized information diets and organized presentations may be useful beyond news for information analysts and other "sensemakers".

Readers and Curators

Our approach is powered by the expertise of curators. Curators or traditional editors set standards for information, both for the quality of sources and the organization of its presentation. In traditional publishing, the number of editors and the scope of subject matter are necessarily limited. Publishers arrange to have enough material to satisfy their audiences and enough curators to vet and organize the material.

We depart from tradition by enabling any user to be a curator, publishing and sharing articles in topically-organized channels. The idea is to reach down the long tail (Anderson 2006) of specialized interests with a growing group of curators. This approach draws on three sources of power that we call the light work of the many (the readers), the hard work of the few (the curators), and the tireless work of the machines (our system).

Supporting Readers and Curators

Our two classes of users, readers and curators, need distinct kinds of support. Readers select subject areas of interest and the system provides current information, vetted and organized. Busy readers want to fit their reading into available moments of their days. Sessions are of variable length and the article presentation should help readers to get the most important information efficiently, whether they have a minute, five minutes, or longer. Our users can access the system using web browsers at their computers or mobile devices. Readers should be able to allocate attention dynamically, getting details or more articles on a topic when it captures their interest. The system should foster information discovery, so that readers can move sideways to related topics and discover new interests.

When articles come from many sources, the main work of curators is finding and organizing them. Automating this work is the main opportunity for supporting curators. Automation requires capturing the relevant expertise of the curators, who are often busy people. A challenge is for the system to acquire their expertise efficiently, enabling them to train the system rather than providing explicit, detailed rules. To support flexibility for subject areas with different needs, the system should enable curators to organize their information in their own folksonomies as in Figure 1.

Figure 1. A topic tree folksonomy for "Future of Journalism".

Application Description

Our application is implemented as a web service (www.kiffets.com). Most of our programming is web and distributed systems ("cloud") technology. About one-fourth is artificial intelligence or information retrieval technology (Jones and Willett 1997).

System Architecture

Figure 2 shows the system architecture. Users access the system through web browsers. Interactivity is provided by browser programs written in HTML/JavaScript and Adobe Flash. The API to the web server uses REST protocols with arguments encoded in JSON. The web server is outside our firewall and uses Django as the web framework. Code for transactions with the rest of the system is written in Python.

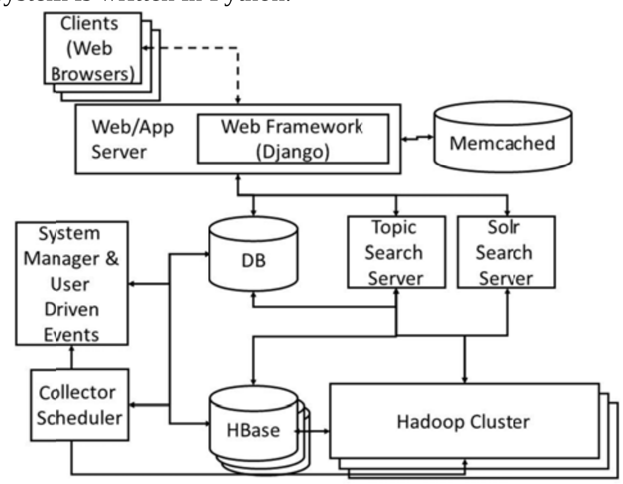

Figure 2. System architecture.

Web and Distributed Systems Technology

Transactions through the firewall are directed to a MySQL data base, a Solr (variant of Lucene) server that indexes articles, and a topic search server that we wrote in Java. These servers are for transactions that require fast, low-latency computations. The computations include user-initiated searches for articles or topics and services for curators who are tuning their topic models and finding new sources. Another server caches query results to reduce load on the database for common queries. All of these servers run on fairly recent mid-class Dell workstations.

Most of the information processing is carried out by Java programs that run on a Hadoop cluster of a dozen workstation-class computers. The main work of the cluster is in crawling curator-specified RSS feeds on the web, collecting and parsing the articles, classifying articles by topic, and clustering related articles from multiple sources. Most of the article information (about 3 terabytes) is stored in HBase, a NoSQL (key-value pair) database that runs over Hadoop's distributed file system. Hadoop also runs other jobs that pre-compute information for the news presentations.

AI Technology for Robust Topic Identification

In manual curation the most time-consuming part is finding and identifying articles for topics. Kiffets classifies 20 to 30 thousand articles by topic every day. Manual curation is practical for traditional newspapers and magazines because the number of topics is small and the articles are drawn from very few sources. Our approach extends curation to a regime of information abundance, where there can be thousands of sources, a proliferation of topics, and where information for narrow and specific topics may be sparse in the sources. Restated, our approach enables systematic curation on the web.

Many online news systems classify articles automatically by matching a user-supplied Boolean query against articles. However, several common conditions can cause this approach to be unsatisfactory. One issue is that common words often have multiple meanings. Does a search for "mustang" refer to a horse, a car, or something else? User expectations of precision are much higher for automatic article classification than for results of search engines. When someone uses a search engine, they face a trade-off between carefully developing a precise query and spending time foraging through the results. In a search setting, it can be acceptable if 50 percent or more of the results are off topic as long as a satisfactory article appears in the top few results. However, readers perceive such imprecision as unacceptable when a system supplies its own query and there are many off-topic articles.

Skilled searchers and query writers can address this issue to a degree by writing more complex queries. We have found, however, that complex queries are prone to errors and refining them is often beyond the skill and patience of our curators.

One way that we have addressed query complexity is by developing a machine learning approach to create optimal queries. In this approach a curator marks articles as on-topic (positive training examples) or off-topic (negative training examples). The system searches for the simplest query that matches the positive examples and does not match the negative ones.

Because we have reported on this approach earlier (Stefik 2008), we describe it here only briefly. Our system employs a hierarchical generate-and-test method (Stefik 1995) to generate and evaluate queries. The queries are generated in a Lisp-like query language and compiled into Java objects that call each other to carry out a match. The articles themselves are encoded as arrays of stemmed words represented as unique integers. With query-matching operations implemented as operations on memory-resident numeric arrays, the system is able to consider tens of thousands of candidate queries in a few seconds. This is fast enough for interaction with a curator.

The query terms are chosen from the training examples, focusing on words that have high TFIDF ratios, that is, words whose frequencies in the training examples are substantially higher than their frequencies in a baseline corpus. The generated query relationships are conjunctions, disjunctions, n-grams, and recursive compositions of these. Candidate queries are scored according to matching of the positive and negative training examples and structural simplicity.

Although the optimal query generator automates writing queries, this approach does not get around fundamental problems with using queries alone to classify articles. For example, it does not distinguish cases where articles match a query incidentally, such as when article web pages contain advertisements or short descriptions provided by a publisher to draw a reader to unrelated articles. From the perspective of article classification, this information on a web page is noise. The query approach also does not distinguish articles that are mainly on-topic from articles that are mainly off-topic, but which contain tangential references to a topic. For this reason, we characterize the query approach as having high precision and high vulnerability to noise.

To reduce noise vulnerability, we incorporate a second approach to classifying articles. The second approach complements query matching and has opposite characteristics. In contrast to the query approach, it has low vulnerability to noise but also low precision.

The second approach considers an article as a whole, rather than focusing on just the words and phrases in a query. It represents an article as a term vector, pairing basis words with their relative frequencies in the article. We compute the similarity of the term vector for an article to a term vector for the topic as derived from its training examples. This is a variant of standard similarity approaches from information retrieval. With a cosine similarity metric, the score approaches one for a highly similar article and zero for a dissimilar article. A similarity score of about 0.25 is a good threshold for acceptability.

In summary, our system combines two topic models with opposite characteristics to provide a robust classification of articles by topic. An article is classified as on-topic if it matches the query for a topic and has a high enough similarity score. This combined method has proven precise enough for topics and robust against the noise found in most articles. It requires that curators identify good examples of on-topic and off-topic articles. The curator knowledge is captured from the training examples that they select. For most topics, three to six training examples of each type are enough for satisfactory results.

AI Technology for Multi-level Topic Presentation

Readers expect articles to be classified and well organized in sections corresponding to topics. For example, in a channel covering hard core national news, there are currently over 300 topics for articles drawn from several hundred sources. The topic tree has eight top-level topics including “Crime and the Courts,”, “Economy and Trade,”, “Health and Safety,” “Politics,” and “War and Terrorism.” Eighty to a hundred articles are collected and classified each day for this channel. Figure 3 gives examples of three articles promoted from subtopics of “Health and Safety”. The first article comes from the leaf topic “Snow”. Its full topic trail is “USA > Health and Safety > natural disasters > Storms > Snow”.

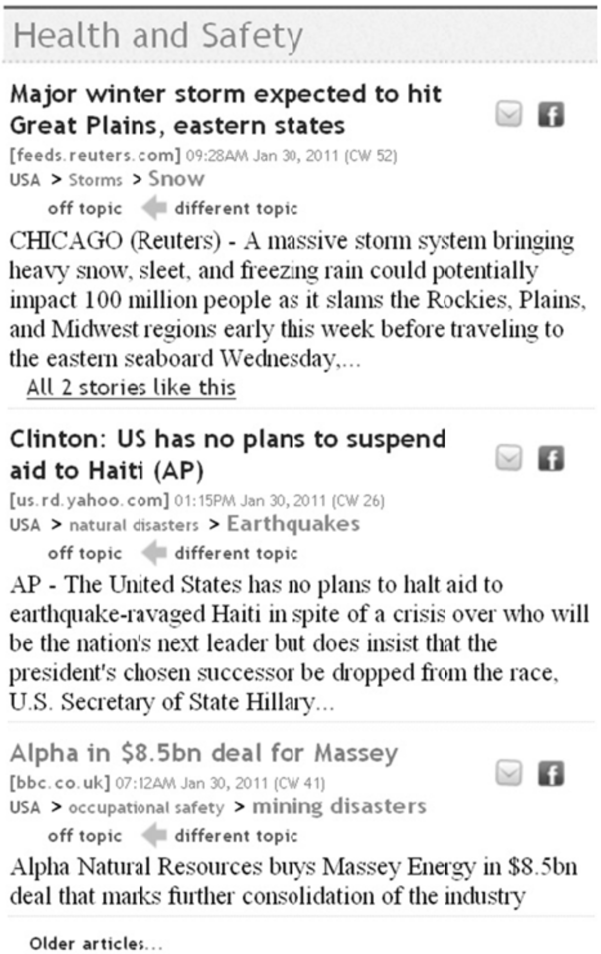

Figure 3. Displaying articles promoted from subtopics.

In an early version of the system, all of the new articles for a channel were presented at once. This was overwhelming for readers even though articles were accurately classified by topic. In a second version, users had to click through the levels of the tree to find articles, which was too much work and caused readers to miss important stories. In the current version, articles are presented a level at a time and a rationed number of articles from the leaf topics are selectively bubbled up the topic tree through their parents. Which articles should be selected to propagate upwards through each intermediate topic? The considerations and AI techniques for this approach are the subject of another paper. For this reason, we describe the approach very briefly here.

Our system combines several factors in promoting articles through levels. Articles are favored if they are central to a topic, that is, if their term vector is similar to the composite term vector for a topic or close to one of its individual training examples. Articles from a topic are favored if the topic is hot, meaning that the number of articles on the topic is dramatically increasing with time. Story coverage in a parent topic is allocated to have some balance across competing subtopics.

These computations are carried out in parallel across all topics using Hadoop as a job scheduler on the back-end of the system with the results saved in the MySQL database. This enables the front end of the system to present the best articles for the particular interests of each user without executing expensive queries in real time for each topic.

AI Technology for Detecting Duplicate Articles

Busy news readers expect a news system to help them to satisfy their interests efficiently. They can be annoyed if duplicate articles appear under a topic. Exact duplicates of articles can be collected by the system when curators include multiple feeds that carry the same articles under different URLs. Reader perception of duplication, however, is more general than exact duplication and includes articles that are just very similar. Similar articles might come together in the same topic or via the article promotion process to high level topics. The challenge is finding an effective and efficient way to detect duplicates.

Our approach begins with simple heuristics for detecting identical wording. The main method uses clustering. Since the number of clusters of similar articles is not known in advance we developed a variant of agglomerative clustering. We employ a greedy algorithm with a fixed minimum threshold for similarity. Two passes through the candidate clusters are almost always enough to cluster the duplicate articles. An example of a clustering result is shown below the first article in Figure 3 in the link to “All 2 stories like this.” In our current implementation, duplicate removal is done only for displays of articles from the previous 24 hours.

Other AI Technology for Information Processing

Most of the programming in the system is for system tasks such as job scheduling, data storage and retrieval, and user interactions. Nonetheless, AI techniques have been essential for those parts of the system that need to embody knowledge or heuristics. Here are some examples:

- A hot-topics detector prioritizes topics according to growth rates in editorial coverage across sources, identifying important breaking news.
- A related-topic detector helps users discover additional channels for their interests.
- A near-misses identifier finds articles that are similar to other articles that match a topic, but which fail to match the topic's query. The near-miss articles can be inspected by curators and added as positive examples to broaden a topic.
- A source recommender looks for additional RSS feeds that a curator has not chosen, but which deliver articles that that are on topic for a channel.

Interweaving Development and Evaluation

This project was inspired by "scent index" research (Chi, Hong, Heiser, Card, and Gumbrecht 2007) for searching the contents of books. That research returned book pages as search results organized by categories from the back-of-the-book index. For example, a search query like "Ben Bederson" in an HCI book returned results organized by topics corresponding to Bederson's research projects and institutional affiliations. We thought it would be exciting to extrapolate from a given index to organize web search results.

The key technological uncertainty was whether a machine learning approach could accurately model index topics. A one-person internal project was started that built and developed the first version of an optimal query generator. After a few months we showed that it could quickly generate queries that accurately matched pages for all 900 index entries in a book, essentially reproducing results of the original index (but finding errors in it).

Alpha and Beta Testing

In April 2008 we created a two-person team to explore the application of this technology. The initial business objective was to create a prototype product suitable for an advertising-based business delivering personalized news. Later the objective evolved to provide information processing services for news organizations.

In October 2008 we opened our first prototype to alpha-testing by a dozen users. We had a flash-based wizard for curators and a simple web interface for readers. Each of the curators built a sample index and used it for a few weeks. Four more people joined the team, focusing on release testing, user interviews, design issues, and fund raising.

Although the system was able to collect and deliver articles when we built the channels, it became clear that curation was too difficult for our first curators. They had difficulty finding RSS feeds and did not completely grasp the requirements of curating. Extensive interviews and observation session helped us to identify key usability issues. We came to understand that the system would not be a commercial success unless it went viral. This required making it much easier to use.

We began learning about lean start-up practices and became obsessed with meeting customer needs. We followed a ruthless development process that divided user engagement into four stages: trying the system, understanding it, being delighted by it, and inviting friends. We divided possible system improvements into a track for curators and a track for readers. We built performance metrics into the system and monitored user engagement with Google Analytics. In 2010 we measured 1300 unique visitors per month with about 8900 page views. The average user stayed for about eight minutes, which was high. Every month we interviewed some users. Every morning we met for an hour to prioritize and coordinate the day's development activities.

The development and deployment of AI technology was driven by the goal of meeting user needs. For example, when article classification began failing excessively due to noisy articles from the web, we combined our symbolic query-based approach with the statistical similarity-based approach. For another example, multi-level topic presentation was developed to improve user experience on big channels. Other additions such as the source recommender were prioritized when they became the biggest obstacles to user satisfaction.

Over time we came to understand user and curator habits more deeply. For example, when we recognized that curators wanted to tune their topic models while they were reading their daily news, we eliminated the separate "wizard" for curators and incorporated curation controls into the news reading interface. This required changing how the machine learning algorithms were triggered. They went from being requested explicitly in a curation session to being requested implicitly when articles were added to topics (positive examples) or when articles were marked as off-topic during reading. We did not always gain our biggest insights through user interviews and metrics. Some of our insights came from being heavy users ourselves.

Performance Tuning

In early 2009 we began beta-testing with about 60 users. The system load from users and articles increased to a level where we had to prioritize scaling and robustness issues. The first version of the system began to stagger when we reached 100 thousand articles. A recurring theme was to reduce the I/O in processes, since that dominated running time in most computations. For example, an early version of the classifier would read in arrays representing articles and use our optimized matching code to detect topic matches. Recognizing that most of the time was going into

I/O, we switched to using Solr to compute indexes for articles when they were first collected. The classifier could then match articles without re-reading their contents.

We switched to a NoSQL database for article contents to support the millions of articles that the system now held. We periodically re-worked slow queries and found more ways to pre-compute results on the back-end in order to reduce database delays for users. In June of 2010, we started an open beta process by which any user could come to the system and try it without being previously invited. By August, the system had over 600 users and was able to run for several months without crashing.

Competing Approaches

At a conference about the future of journalism, Google's Eric Schmidt spoke on the intertwined themes of abundance and personalization for news (Arthur 2010).

> The internet is the most disruptive technology in history, even more than something like electricity, because it replaces scarcity with abundance, so that any business built on scarcity is completely upturned as it arrives there.

He also reflected on the future of mass media and the news experience.

> It is … delivered to a digital device, which has text, obviously, but also color and video and the ability to dig very deeply into what you are supplied with. … The most important thing is that it will be more personalized.

There is little question that the news industry is being disrupted and that news is currently abundant. However, although it is appealing to busy people, at the time of this writing we know of no big commercial successes in personalizing news. That said, many news aggregation and personalization services have appeared on the web over the last few years. Some of these services have been popular, at least for a while. In the following we describe the elements that are similar or different from our approach.

Choosing Who to Follow

A few years ago RSS ("Really Simple Syndication") readers were introduced to enable people to get personalized news. RSS readers deliver articles from RSS feeds on the web, created by bloggers and news organizations. RSS readers do not organize news topically and do not provide headlines of top stories. Rather, they display articles by source. A news consumer can read articles from one source and then switch to read articles from another one. Different reader systems vary in whether they are web-based or computer applications and in how they keep track of the articles that have been read. According to a 2008 Forrester report (Katz 2008), however, consumer adoption of RSS readers has only reached 11%, because people do not understand them.

The Pew Internet & American Life Project studies changes in how people consume and interact with news. Much of the growth in online services with news is in systems like Twitter and Facebook, which are similar to RSS readers in that users specify their interests in terms of sources or people that they want to follow. According to Pew, internet sources have now surpassed television and radio as the main source of news for people under 30.

Matching Key Words

News alert systems ask users to provide key words or a query that specifies the news that they want. This approach treats personalization as search. Typical users receive news alert messages in their email.

Since news alert system maintain a wide spectrum of sources, they sidestep the problem of asking users to locate or choose appropriate RSS feeds on the web. However, a downside of using a broad set of sources to answer queries is that many of the articles delivered are essentially noise relative to the user's intent, due to unintended matches to incidental words on the web pages containing the articles.

Another disadvantage of news alert systems is that the precision of queries inherently limits their potential for surprise and discovery. In struggling to get just the right query, news consumers potentially miss articles that express things with different words. Furthermore, news consumers want to find out about what's happening without anticipating and specifying what the breaking news will be.

Personalized News by Mainstream Publishers

Some major news publishers let their customers choose from a pre-defined set of special interest sections such as (say) "Science and Technology" or allow them to specify key words that are matched against news articles from the publisher. The pre-defined sections are manually curated, and the key word sections rely on simple matching. According to a private communication from a technology officer of a major national news publisher, fewer than three percent of their mainstream news customers enter any form of customizing information.

Systems like Google News offer a similar combination of methods except that they draw from many sources. They offer predefined channels (World, Business, Sci/Tech) on broad topics which seem to achieve topical coherence by showing only articles from appropriate manually-curated feeds. Any user-defined channels based on key words have the same noise problems as other key word approaches. Google News also uses a clustering approach to identify hot articles. Lacking sections defined by topic trees, it does not organize articles into coherent, fine-grained sections. These services are simpler to use than RSS readers because users need not select sources.

Collaborative Filtering

Besides these main approaches for personalized news, there are also social approaches for gathering and delivering news. Collaborative filtering approaches recognize that "birds of a feather" groups are powerful for recommending particular news (and movies, books, and music). These systems collect data about user preferences, match users to established groups of people with similar interests, and make recommendations based on articles preferred by members of the groups. Findory and DailyMe are examples of early and current news systems, respectively, that use collaborative filtering to deliver personalized news.

The affinity groups for users need to be identified. Identification can be done explicitly by asking users to rank their interests in a questionnaire. It can also be done implicitly by keeping track of articles that users read. Since people typically have several distinct news interests, each interest has to be separately accounted.

By itself, collaborative filtering provides no means for organizing information in topical sections. Restated, there is no topic tree or topical organization beyond the umbrella category for a group.

Some news sites based mainly on other methods use collaborative filtering to support a degree of personalization. These systems keep track of stories that users click on and use collaborative filtering to identify and predict personalized interests of the readers. Stories matching the personalized categories are promoted to higher prominence in the presentation.

Social News Sites

Social news sites such as Reddit or Digg enable people to submit articles. The articles are ranked by popularity according to reader votes. Social bookmarking sites such as Delicious are near cousins to social news sites. Their primary purpose is to organize a personal set of browser bookmarks to web pages, and their secondary purpose is to share and rank the bookmarks. Social news sites rely on social participation both for collecting and ranking articles and with enough participants can address topics on the long tail.

Although social news sites sometimes have specialized channels to personalized interests, there is a challenge in getting an adequate stream of articles for narrow topics, especially when the participating groups are just getting established. Furthermore, the presentation of news on social news sites is not topically organized and usually appears quite haphazard because articles are listed by popularity without topical coherence.

Conclusions

This project was inspired by another project that returned pages as search results from a book organized by its index. To generalize that concept to the ever-expanding web, we needed to develop an effective method for extrapolating from an explicit index over a fixed corpus to a model-based evergreen index. Figure 4 gives an example of our system doing this when this paper was written.

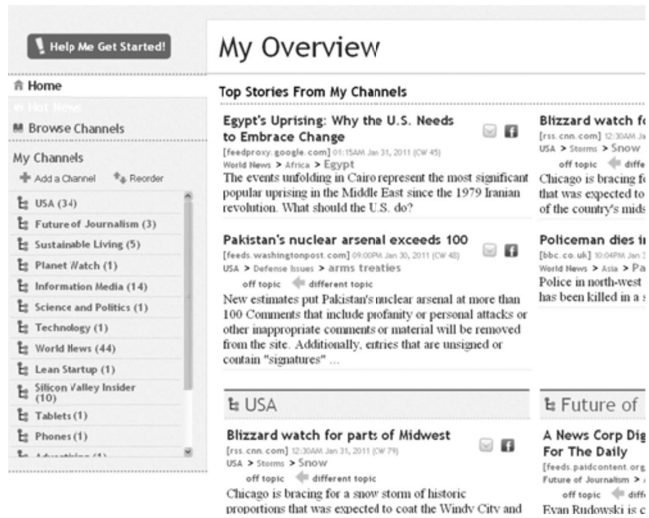

Figure 4. Overview page of reading interface.

The figure shows a set of curated channels selected by the user. Users can subscribe to channels that cover mainstream subject areas like the channels "USA" or "World News" in this example. They can also subscribe to channels in more specialized areas, such as "Sustainable Living" or "Future of Journalism". Given enough time, they can scroll down the page to see top articles from each of the channels. For more, they can drill down selectively to get more topics and more articles.

We did not start out with a goal of using artificial intelligence technology. Rather we used the technology of choice at each stage of trying to satisfy user requirements.

Our system follows the "knowledge is power" logic of earlier AI systems on knowledge systems in that it depends on the expertise of its curators. We acquired curator expertise using a machine learning approach where curators can select sources that they trust (sometimes guided by source recommendations from the system), organize topics in a topic tree folksonomy according to how they make sense of the subject matter, and train the topic models with example articles. Using this information the system automatically creates evergreen channels of information for readers every day.

A major challenge was in making curation easy and reliable given the limited time that curators have available. It is easy for us to train curators in a couple of short sessions. It is more challenging to attract people on the web to try the system, to understand what it does, and to invest in becoming a curator.

During our trial period about one registered user in three created a channel and about one in four of those created a complex channel. We do not know ultimately what fraction of users might be expected to become curators. Many users create very simple channels without bothering to set up topics. We believe that there are very interesting pivots to make on new mobile devices and in engagements with online communities. There are also other applications of the classification technology beyond personalized news.

Further development on this project depends on finding external funders or investors. The news business is increasingly undergoing rapid change and economic challenges. It is changing on several fronts, including how news is delivered (mobile devices), how it is being reported (citizen journalists and content farms), how it is paid for (subscription services, pay walls, and advertising). This project opens a further dimension of change: how abundant news can be curated.

Kiffets was designed, implemented, and deployed by two people over two and a half years. Other project members worked on evaluation, channel development, user experience, release testing, and business development.

Acknowledgments

This research was sponsored by the Palo Alto Research Center. We thank Barbara Stefik, Ryan Viglizzo, Sanjay Mittal and Priti Mittal for their contributions to design, release testing, user studies and great channels. Thank you to Lawrence Lee and Markus Fromherz for their guidance and support. Thank you to Jim Pitkow for advice on lean management. Thanks to our Kiffets users for much feedback.

References

Anderson, C. 2006. *The Long Tail: Why the Future of Business is Selling Less of More.* New York, New York.: Hyperion.

Arthur, C. 2010. Eric Schmidt talks about threats to Google, paywalls, and the future. *Guardian*. London, England: Guardian News and Media Limited. (Online at http://www.guardian.co.uk/media/2010/jul/02/activate-eric-schmidt-google)

Chi, E. H., Hong, L., Heiser, J., Card, S. K., and Gumbrecht, M. 2007. ScentIndex and ScentHighlights: Productive reading techniques for conceptually reorganizing subject indexes and highlighting passages. *Information Visualization*, 6 (1): 32-47.

Jones, K. S., Willett, P. 1997. *Readings in Information Retrieval.* San Francisco, CA.: Morgan Kaufmann Publishers.

Katz, J. 2008. What's Holding RSS Back: Consumers Still Don't Understand This Really Simple Technology. Cambridge, Ma.: Forrester Research. (Online at http://www.forrester.com/rb/Research/whats_holding_rss_back/q/id/47150/t/2)

Pirolli, P. 2007. *Information Foraging Theory: Adaptive Interaction with Information.* Oxford, England: Oxford University Press.

Simon, H. 1971. Designing Organizations for an Information-Rich World. In *Communications and the Public Interest*, ed. Martin Greenberger. 37-72. Baltimore, MD.: The Johns Hopkins Press.

Stefik, M. 2008. Social Indexing. *CHI 2008 Sensemaking Workshop*. April 5-10, 2008, Florence, Italy.

Stefik, M. 1995. *Introduction to Knowledge Systems.* San Francisco, CA.: Morgan Kaufmann Publishers.