Design and Deployment of a Personalized News Service

Mark Stefik, Lance Good

■ *From 2008–2010 we built an experimental personalized news system where readers subscribed to organized channels of topical information that were curated by experts. AI technology was employed to present the right information efficiently to each reader and to reduce radically the workload of curators. The system went through three implementation cycles and processed more than 20 million news stories from about 12,000 Really Simple Syndication (RSS) feeds on more than 8000 topics organized by 160 curators for more than 600 registered readers. This article 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). Traditional mainstream media cover general interest news to attract a large audience. 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 covered there. Satisfying all of our information needs requires visiting multiple sites, but this requires navigation and there is often much overlap in news coverage across sites. The goal of our system (called Kiffets) is to provide readers with news at a single site that is personalized, curated, and organized.

Approach to Personalizing News

A snapshot of a personal information appetite or "information diet" reveals further nuances. Some interests endure. 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.

Kiffets readers explicitly select channels to cover subject areas that interest them. The list of My Channels on the left in figure

  ISSN 0738-4602

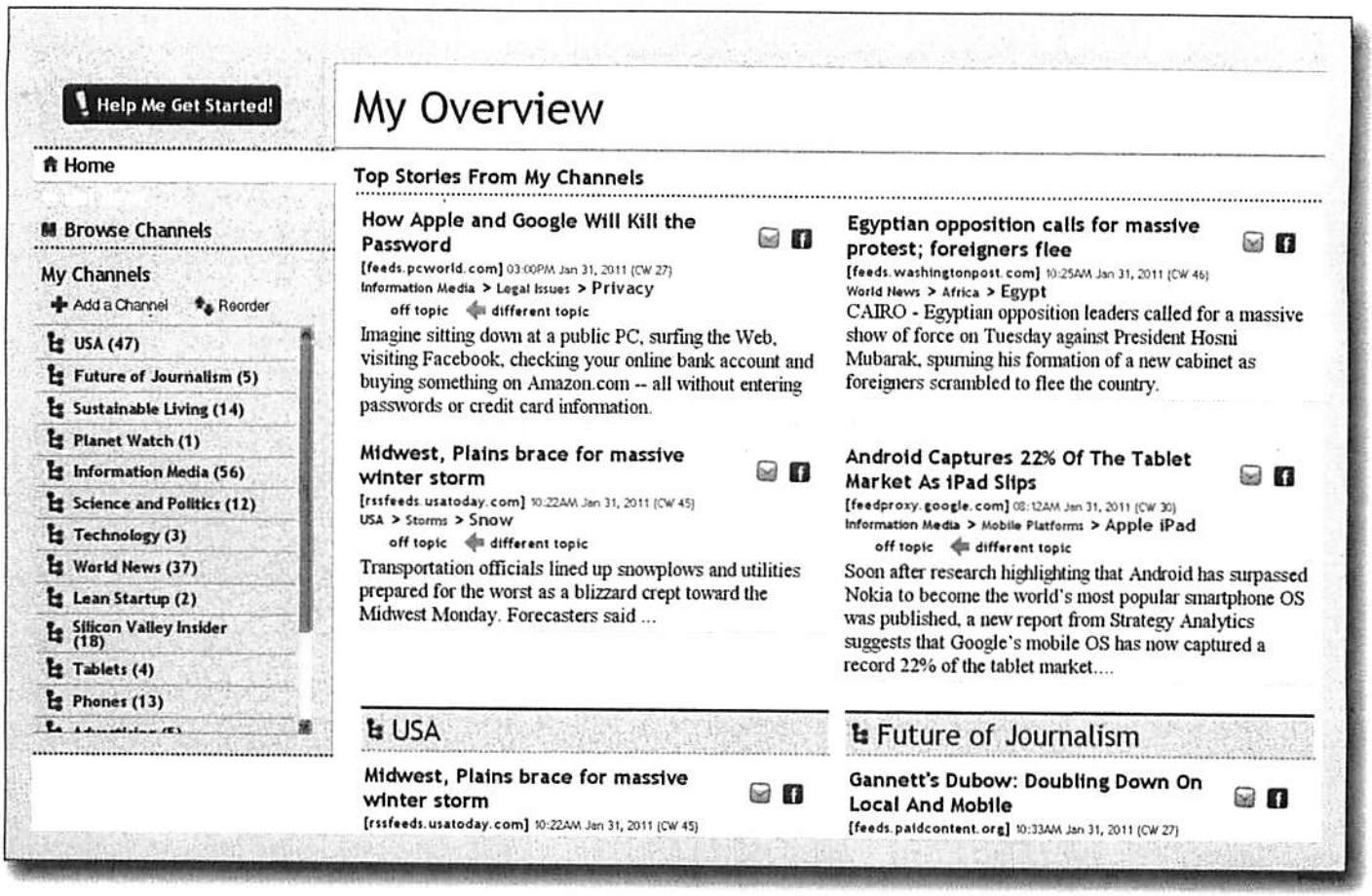

Figure 1. A Reader's Overview of Subscribed Channels for News.

1 shows a list of the channels to which this reader subscribes including USA, The Future of Journalism, Sustainable Living, Planet Watch, Science and Politics, and several more.

The information on the right under My Overview shows the top stories from these channels. The top four stories included a Privacy story from the Information Media channel, a story about Snow from the USA channel, a story about Egypt from the World News channel, and a story about the relative market share of tablets from the Information Media channel. The subsequent presentation shows top stories for each of the channels, organized by topic.

To provide this reader experience, a news delivery system needs to use knowledge from news editors. It needs to collect articles from appropriate sources, classify them, and organize them by topic. Creating this organized presentation of news automatically requires capturing and harnessing the relevant subject matter expertise. In traditional news publications, editors use their expertise to select sources and organize news presentations. Publishers arrange to have enough material to satisfy their audiences and enough editors to vet and organize the material.

We call our corresponding subject matter experts curators. We depart from tradition by enabling any user to be a curator and to attract a following by sharing articles in organized, topical channels. We also simplify curation by automating the repetitive work. The idea is to extend topical coverage down the long tail (Anderson 2006) of specialized interests by using a large group of curators.

What Readers and Curators Need

Based on interviews with Kiffets users, busy readers want to forage for news efficiently. A "5-20-60 rule" expresses how a reader with a busy lifestyle allocates reading time.

> **5-20-60 rule.** I typically take time to read 20 articles a day and scan 60. When I'm in a hurry I only have time to read 5 articles according to my interests. When I have time I want to drill down selectively for more information.

Such busy readers want to optimize their use of available time to follow the news. They want to read the important stories first and to scan the rest of the news. Although they want to be informed about topics at the edges of their interests, they have a strong sense of priority topics that they

want to keep up on. They want adequate cues or "information scent" so that they can tell whether a given story is interesting. They drill down accordingly as their interests dictate and they have time. Such actions constitute information foraging strategies (Pirolli 2007).

In Kiffets, readers leverage the knowledge of curators. A curator has a strong interest and expertise in a subject area. Curators want to share their perspective with others. They are not paid for their work and depend on Kiffets to do the heavy lifting of collecting, classifying, and presenting news. They are not programmers and typically have no understanding of AI or machine-learning technology.

Our goal was to provide curators with a simple and intuitive interface for selecting trustworthy news sources and creating a useful topic structure for organizing the news. They need a simple and intuitive way of conveying their understanding of how to classify articles into topics.

In this way our approach draws on three sources of power that we call the light work of the many (the readers), the harder work of the few (the curators), and the tireless work of the machines (our system).

Guide to Reading

Most of the technology supporting our personalized news service is web and distributed systems (cloud) programming. About one-fourth is artificial intelligence and information retrieval technology (Jones and Willett 1997). This includes mainly topic modeling and machine learning technologies that we developed.

The next two sections describe the Kiffets personalized news service first as a software-as-service application built to meet user requirements, and then in terms of its underlying AI technology. Later sections describe competing approaches and lessons learned.

Application Description

Manual curation is practical for traditional newspapers and magazines because the topics corresponding to newspaper sections are broad, their number is small, and the published articles are drawn from just a few sources.

Personalized news differs from traditional mainstream news because it requires a regime of information abundance. Some narrow topics are very infrequently reported, appearing sparsely in a range of publications. Topics of general interest are often widely reported, with similar articles appearing in many publications.

In 2010 Kiffets curators used more than 12,000 RSS feeds as sources for news. These feeds contributed between 25,000 and 35,000 articles every day, which were classified automatically by topic. The bigger channels on our system used a few hundred feeds as sources and could pull in hundreds of articles per day for their topics.

Kiffets employs a distributed computing architecture to deliver personalized news rapidly to readers as they explore their interests. It also gives quick feedback to curators as they define topic folksonomies and tune individual topic coverage by selecting training examples. Most of the information processing of news is done continuously by a collection of backend processors, caching results for delivery by a web server.

Supporting Readers

Readers select subject areas of interest and the system provides current information, vetted and organized.

Finding Relevant Channels

Suppose that a reader wants to explore information about Egypt, going beyond the channels to which the reader has already subscribed. Figure 2 shows the results of searching for channels about Egypt when a draft of this article was written. The results show the most relevant channels, together with their most relevant subtopics and a sample article.

The search query "Egypt" does not specify what aspect of that country the reader is interested in. The delivered results show channels that focus variously on Egypt as a travel destination, Egypt in the context of recent World Cup soccer, Egypt in world news, and so on.

Figure 3 shows the results of clicking the first search result to the Egypt channel. The display offers articles from different time periods, such as the previous 24 hours, the previous 2 days, and so on.

To guide the reader in further browsing, the presentation links to related topics from the Egypt sightseeing channel including Cairo Sightseeing and Archeological Finds in Egypt. A reader interested in traveling to Egypt can check out the Egypt sightseeing channel. A reader more interested in the Egyptian political situation could visit one of the curated channels specific to that interest.

As readers' interests shift they can subscribe or unsubscribe to curated channels or create new channels themselves.

Making Foraging Efficient

Most users subscribe to a few channels — some on general news and some on specialized topics. Showing all of the articles on the front page would present an unwieldy flood of information.

Kiffets presents top articles as starting points and enables readers to drill down selectively. Figure 4 shows articles that appear if a reader drills down at

Health and Safety in the USA folksonomy.

Readers particularly interested in natural disasters can drill down again to see articles about earthquakes, fires, storms, volcano eruptions, and other natural disasters as in figure 5. Each drill down reveals more topics and more articles. This novel Kiffets capability is enabled by AI technology described later. It supports information foraging and is one of our favorite features.

Supporting Curators

Curators are often busy people. Like traditional news editors, curators select reliable sources and decide how the articles should be organized. In contrast to traditional news organizations, curators on our system do not do this work manually every day. They expect the system to acquire their expertise and to allow them to tune it over time.

When articles come from many sources, the main work is finding and organizing them. Automating this work is the main opportunity for supporting curators.

Figure 6 shows the folksonomy of topics for the channel Future of Journalism. The curator for this channel organizes the topics and indicates training examples for each topic.

System support for curators is enabled by AI technology described later. Besides the automatic classification of articles by topic, it includes machine learning of topic models, hot topic detection, clustering methods to detect duplicate articles, source recommendation, and automatic selection and allocation of space for displaying top articles.

System Architecture

Figure 7 shows the system architecture. Users access the system through web browsers. Browser programs written in HTML/JavaScript and Adobe Flash provide interactivity. 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.

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

Java programs running on a back-end Hadoop cluster of a dozen workstation-class computers carry out most of the information processing. The collector-scheduler process periodically schedules jobs to crawl curator-specified RSS feeds on the web, collect and parse articles, classify them by topic, and cluster related articles from multiple sources. Other periodic Hadoop jobs use AI technology to remove duplicates for topics, identify hot topics, and identify top articles. Still other Hadoop jobs for machine learning are triggered when curators mark articles as on topic or off topic.

Most of the article information (about 3 terabytes) is stored in HBase, a NoSQL (key-value pair) database that runs on Hadoop's distributed file system.

Figure 2. Finding a Good Channel about Egypt.

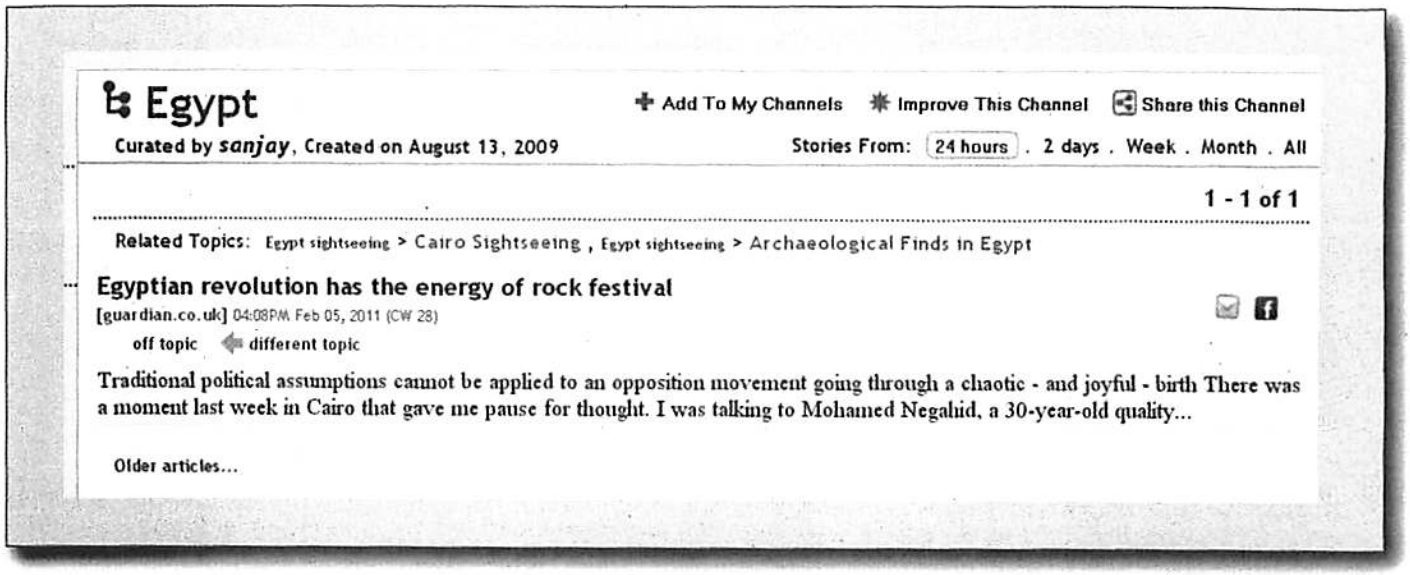

Figure 3. A Channel about Egypt.

Figure 4. Subtopics and Articles for Health and Safety.

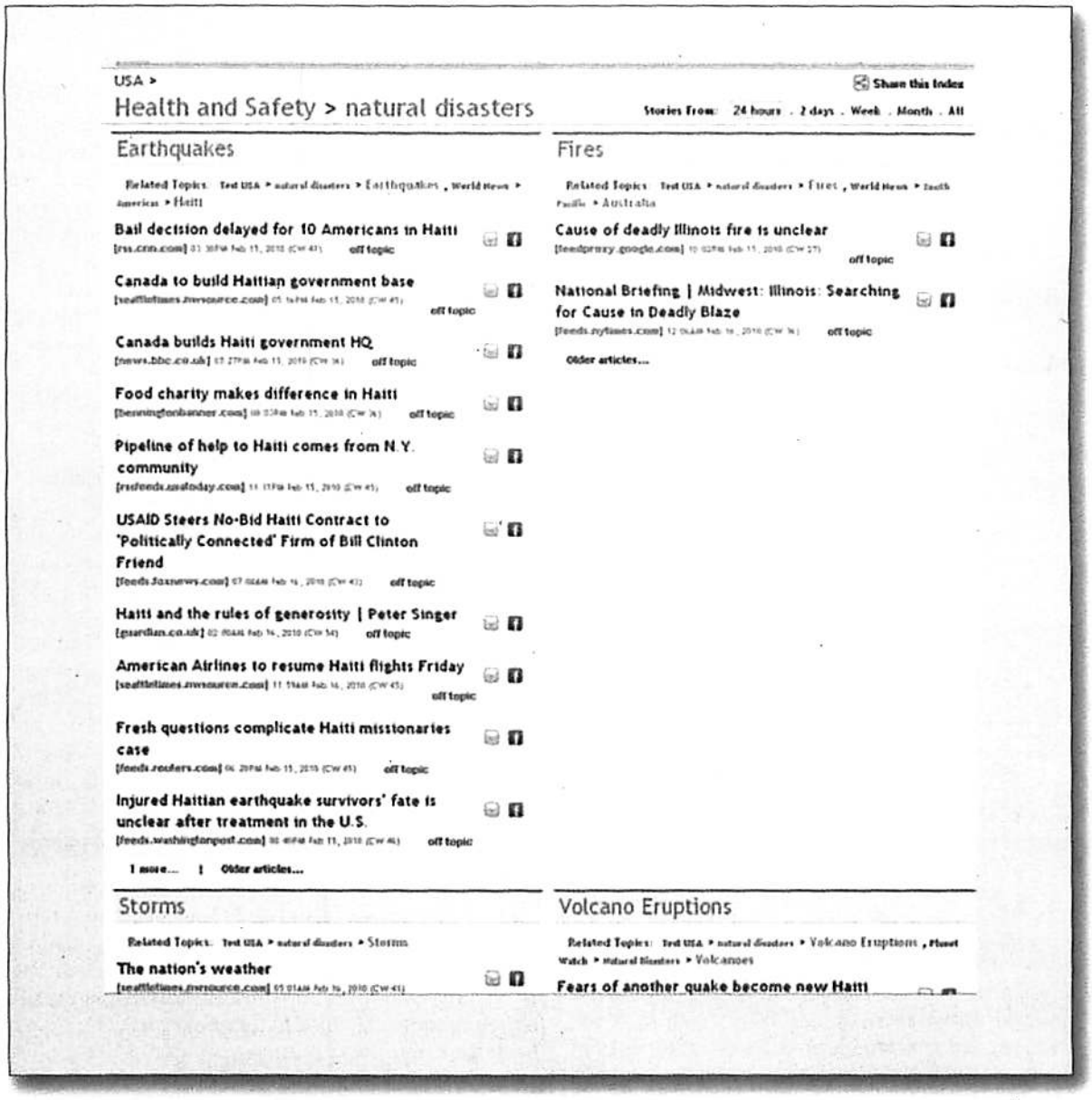

Figure 5. Articles for Different Kinds of Natural Disasters.

AI Technology Description

We used several kinds of AI technology to address the information-processing challenges. The main functions of the AI technology are classifying articles into topics and creating an effective news presentation.

Robust Topic Identification

Accuracy in the topical classification of articles is difficult because there is variability in the words used in articles on a topic and "noise" in news articles. The noise in an article includes advertisements for products and blurbs inserted by publishers to keep readers engaged.

Optimal Query Generation

Many online news systems classify articles automatically by matching a Boolean query against articles. 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 people search interactively, they face a trade-off between carefully developing a precise query and spending time foraging through the results. It is acceptable if many of the results are off topic as long as a satis-

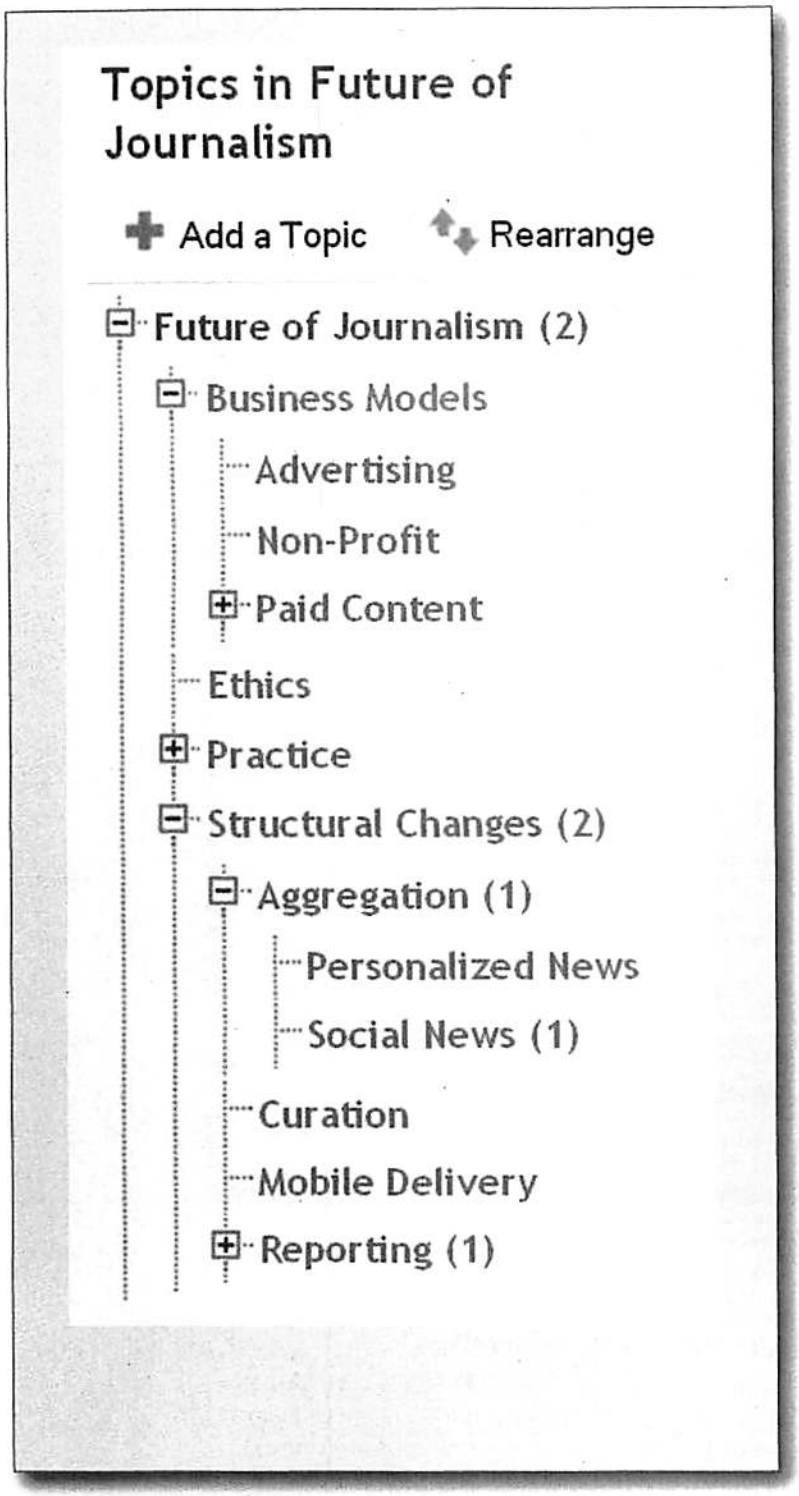

Figure 6. A Topic Tree Folksonomy for Future of Journalism.

factory result appears in the top few. In contrast, users find it unacceptable when a system supplies its own query and there are many off-topic articles.

Skilled query writers can address this issue by writing 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. In practice few people write queries with more than two or three key words and seldom express Boolean constraints.

We developed a machine-learning approach to generate optimal queries. As curators read the news, they can mark articles that they encounter. If they flag an article as off topic, it becomes a negative training example for the topic. If they come upon a particularly good article in their reading, they can recommend it — making it an on-topic training example. Upon receiving new training examples, Kiffets schedules a training job to find a new optimal query and reclassify articles for the topic. Curators need not understand how this works.

Because we have reported on this approach elsewhere (Stefik 2011), 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 expressed in a Lisp-like query language and compiled into Java objects that call each other to carry out a match. The articles 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 several tens of thousands of candidate queries for a topic in a few seconds. This is fast enough to support an interactive session when a curator wants to tune a topic.

The query terms are chosen from the training examples, drawn from words that have high term frequency–inverse document frequency (tf-idf) ratios, that is, words whose frequencies in the training examples are substantially higher than their frequencies in a baseline corpus. The query relationships are conjunctions ("gas" AND "pollution"), disjunctions ("army" OR "navy"), *n*-grams (sequence of words), and recursive compositions of these. Candidate queries are scored, rewarding matches of on-topic examples, penalizing matches of off-topic examples, and rewarding query 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. 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.

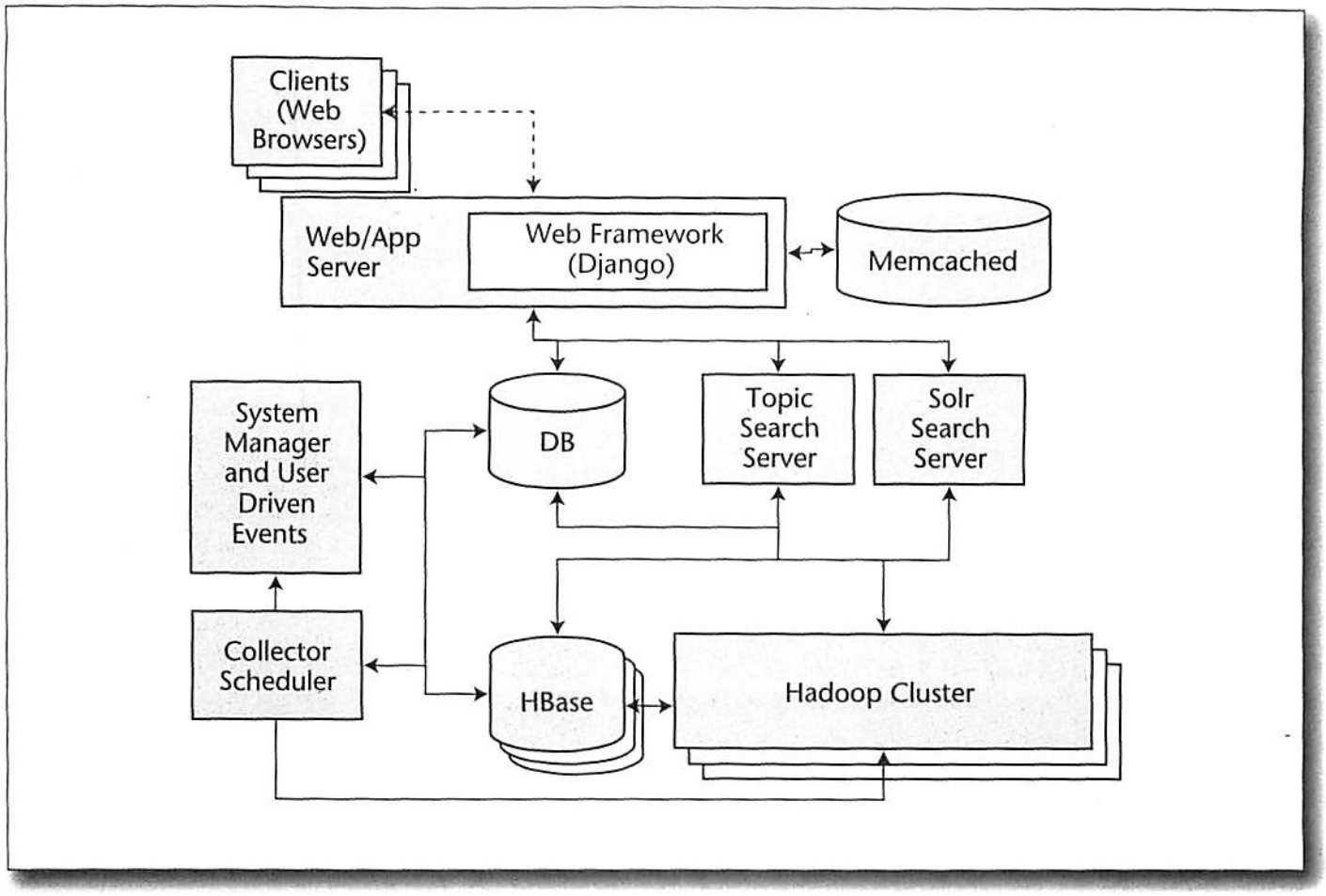

Figure 7. System Architecture.

Single-Core Topics

To reduce noise vulnerability, we incorporate a statistical modeling approach for classifying articles. This 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 statistical 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 (Salton and Buckley 1988), pairing basis words with their relative frequencies in the article. We compute the similarity of the term vector for an article to the term vectors for the topic as derived from its training examples. 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.

For example, with the current concerns about energy and the economy, stories about gas prices often appear in the news. Although some stories are simply about the rise or fall of prices, other stories mention gas prices in relation to other matters. For example, stories about the expenses of living in the suburbs sometimes mention rising gas prices as a factor. Ecological stories about offshore or arctic drilling for oil sometimes predict the effects of increased regulation on gas prices.

Suppose that a curator wants to include articles that are mainly about the rise or fall of gas prices. The curator may be willing to include articles mentioning arctic drilling, but not if they are only incidentally about gas prices. This is our simplest kind of topic model. We call it a single-core topic because it has a single focus. (Multicore topics have multiple foci — such as if a curator specifically also wants to include gas price stories about arctic drilling for petroleum.)

Figure 8 illustrates how we model single-core topics. The outer box surrounds the articles from the corpus. The dashed box surrounds articles that match a Boolean query. The query could be as simple as the terms "gas price" with its implied conjunction or it could be more complex, such as "(gas OR gasoline) AND (price OR cost)." The dark circle

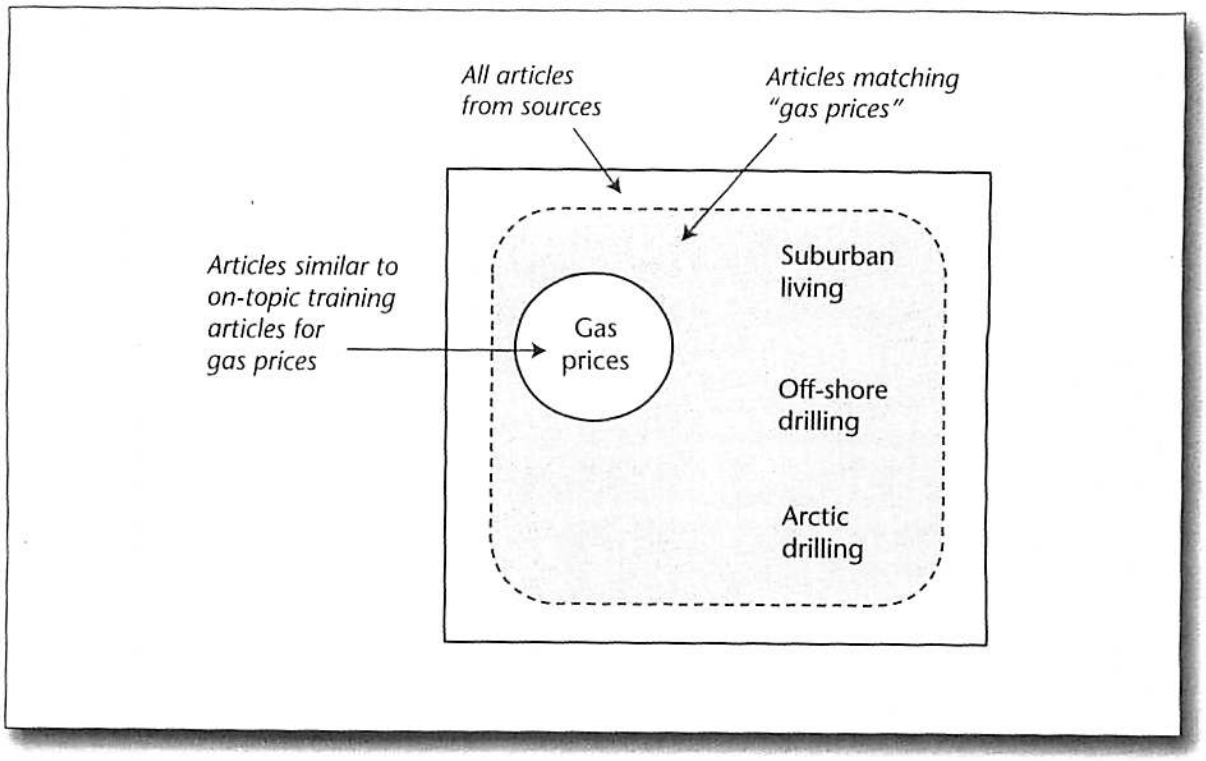

Figure 8. A Single Core Topic for Gas Prices.

represents articles whose similarity to the positive training examples is above a threshold.

A reference vector is computed for each of the training examples. In this example, a reference vector for suburban living would include words describing living far from cities and commuting to work. Reference vectors give a computed tf-idf weight for each word and are the same for either on-topic or off-topic examples.

This combined topic model joins an optimal Boolean query with term vectors for the on-topic and off-topic examples. An article is classified on topic if it matches the query, its vector is similar enough to an on-topic reference vector, and it is not too similar to an off-topic reference vector. This approach combines the high precision of a Boolean query with graduated measures of similarity to training examples. This approach has proven precise enough for topics and robust against the noise found in most articles. Restated, this approach is much less vulnerable to false positive matches to an irrelevant advertisement on a page than query matches alone. For most topics, three to six training examples of each type are enough for satisfactory results.

Multiple-Core and Multiple-Level Topics

Consider a western newspaper that has a section of stories for each state. Suppose that the curator wants to include articles about the Oregon state

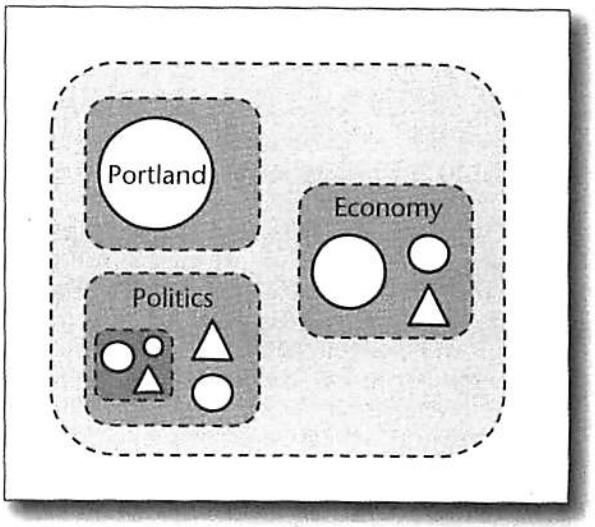

Figure 9. A Multilevel, Multicore Topic Model for Oregon.

economy covering lumber and high technology, but not tourism. For Oregon politics she wants to cover politics in the state capital, politics of the Portland mayor's office, and some controversial state initiatives. She may also want to cover major news from Portland, the state's biggest city.

Figure 9 suggests how the approach for modeling topics works recursively. Dashed boxes repre-

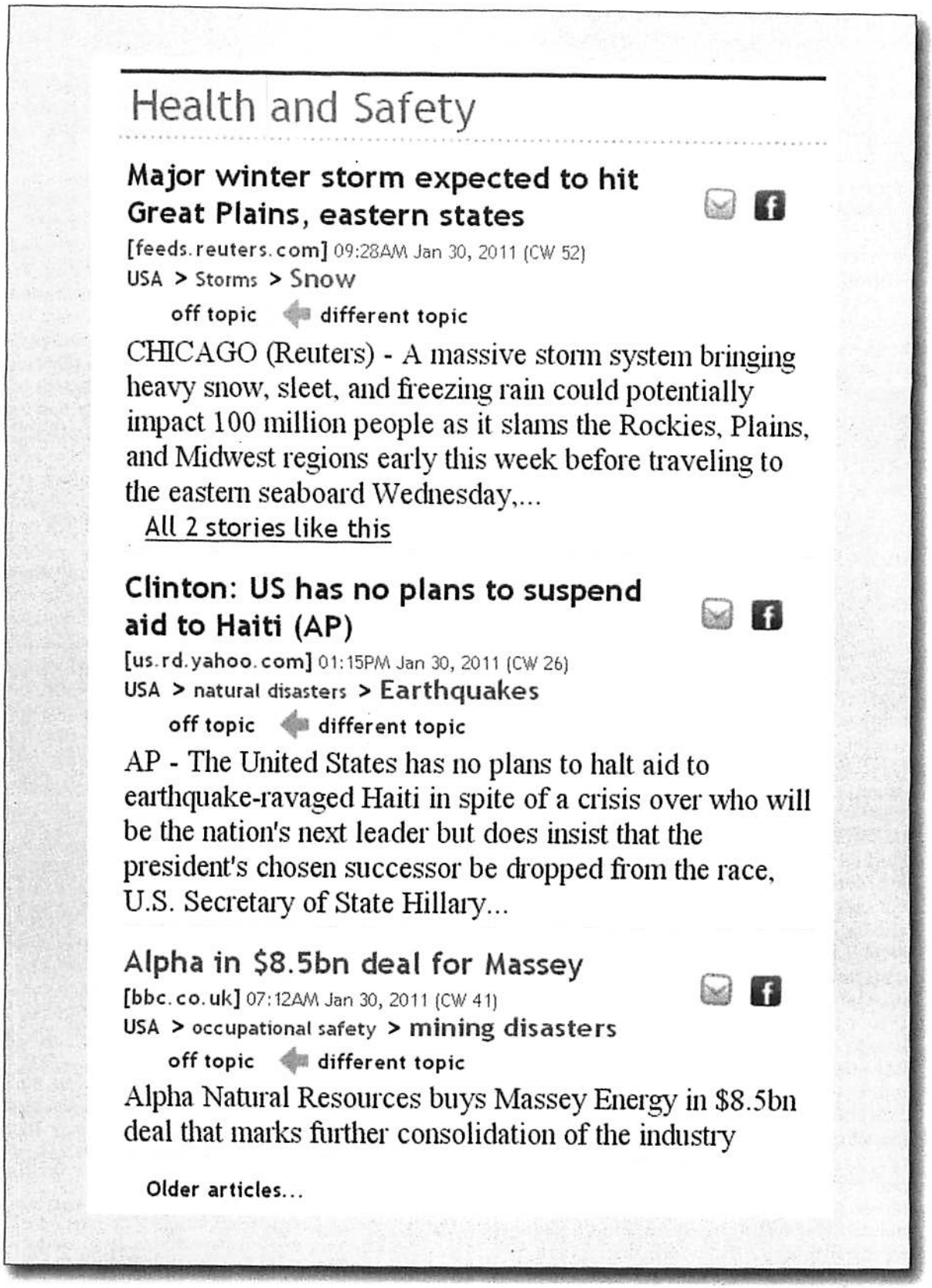

Figure 10. Displaying Articles Promoted from Subtopics.

sent a set of articles defined by a query. Circles represent term-vector filters for including on-topic examples. Triangles represent filters for excluding off-topic examples.

Each of the Oregon subtopics is complex. We call these fractal topics in analogy with fractal curves like coastlines that maintain their shape complexity even as you look closer. The subtopic for Oregon Economy has its own on-topic and off-topic cores shown as small circles and triangles, respectively. When you look closer, the subtopic for Oregon Politics also has multiple levels.

Prioritizing Articles

In an early version of the system, all of the new articles for a channel (over its entire folksonomy of topics) were presented at once. This was overwhelming for readers even though articles were classified by topic. In a second version, we showed only the articles from the top-level topics and readers had to click through the levels of the folksonomy to find articles on deeper topics. This was too much work and caused readers to miss important stories. In the current version, articles are presented a level at a time but 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? Kiffets 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 a 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 balances competing subtopics.

Articles can be classified as matching more than one topic. For example, in the Kiffets USA index, articles about difficulties between Google and China a few years ago were classified under a topic relating to trade, a topic about censorship, and a topic relating to cyber attacks. When top articles bubble up a folksonomy, similar or identical articles might appear from more than one of its child topics. The allocation algorithm presents an article only once at a level showing the topic that matched it best. If the reader descends the topic tree, the article may reappear in a different context. Figure 10 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.

Detecting Duplicate Articles

Busy newsreaders are annoyed by duplicate articles. Exact duplicates of articles can arise 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. 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 10 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 Kiffets 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 that deliver articles that that are on topic for a channel.

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.

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 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. Systems vary in whether they are web based or desktop 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 percent, because people do not understand them.

The Pew Internet and American Life Project reports on 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. Kiffets benefited from the RSS work since it made available the feeds for news feeds and blogs.

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 systems 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 predefined 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 predefined 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 3 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 keyword 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 (www.findory.com) and DailyMe (www.dailyme.com) are examples of early and current news systems, respectively, that use collaborative filtering to deliver personalized news. (*AI Magazine* ran an issue with several articles [Burke, Felfernig, and Göker 2011] on recommender systems.)

Collaborative filtering systems need to identify affinity groups, for example, by explicitly asking users to rank their interests in a questionnaire. Systems can also keep track of the articles that users read and infer groups implicitly. Since people typically have several distinct news interests, systems must account for each interest separately.

Some news sites use collaborative filtering to support personalization. These systems keep track

of stories that users read 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. One of our unaddressed goals for Kiffets was to augment our user modeling and article ranking by collecting individual user data.

The NewsFinder system (Dong, Smith, and Buchanan 2011), which distributes news stories about artificial intelligence, collects and displays explicit user ratings for articles. Although that system is not designed for personalized news or open curation, it does employ some similar AI methods such as for detecting duplicates.

Social News Sites

Social news sites such as Reddit (www.redddit.com) or Digg (www.digg.com) enable people to submit articles. The articles are ranked by popularity according to reader votes. Social bookmarking sites such as Delicious (www.delicious.com) 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 of the users' specialized interests.

Social news sites face 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.

In summary, the advent of RSS feeds has provided the basis for many groups to explore new ways to deliver news. Some technology elements such as key-word matching, tf-idf matching, and clustering have been used in many approaches. Kiffets explored new territory in its automated assistance to curation, its organization of articles in deep folksonomies, and in its robust topic models that combine symbolic and statistical methods.

Lessons Learned

Kiffets was inspired by "scent index" research (Chi et al. 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 were inspired by how the system employed the organization of an index to help a user to tune a query. The organization of pages by index topic created a sense of conversation, informing the user about expert options for seeking more information. We thought it would be exciting to extrapolate the approach to the web.

Research on the project was internally funded at the Palo Alto Research Center (PARC). Our audacious goal was to develop a commercially viable product in a year. Our runway was longer than that. 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. About a dozen patent applications were filed on the technology.

In the course of this work we learned lessons about both business and technology. On the business side, we pursued two approaches for commercialization. One was to create an advertising-supported personalized news service. A second was to create a back-end platform as a service for news organizations. Although we had serious negotiations with two of the top five news organizations in the United States for several months, in the end PARC did not close a deal. The particulars of that are beyond the scope of this article, but PARC has since adapted its strategies to address patent indemnification and service levels. On the approach of creating a stand-alone business, it is worth noting that the news organizations that have grown and succeeded over the same time period either had an entertainment focus or a focused brand for a specific kind of news. (We did not even have pictures on our news pages!) There is little question that the news industry is being disrupted and that news is currently abundant. However, although personalized news seems appealing to busy people, at the time of this writing we know of no personalized news services that have become large and sustainable businesses. In 2008 when we sought investor funding, investors were pulling back from this area. In this approach it is important to develop a viral business where the user base grows very quickly. During our two-year runway, we did not find a key for that in our work.

Alpha and Beta Testing

A major lesson for us was the importance of adopting a lean startup approach (Ries 2011). The key to this approach is in trying ideas quickly with customers, rapidly pivoting to new alternatives. We came to understand the rhythm of a customer-development cycle better as the project proceeded. In the following we describe the interweaving of development and evaluation that we carried out.

In April 2008 we created a two-person team to explore the application of this technology. 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.

A major challenge was in making curation easy and reliable given the limited time that curators have available. Although the system was able to collect and deliver articles when we built the channels, 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 sessions helped us to identify key usability issues.

Over time we came to understand reader 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 interface for curators and incorporated curation controls into the news-reading interface. This required changing how the machine-learning algorithms were triggered. We shifted from having curators request topic training explicitly to triggering it implicitly by marking articles while they were reading.

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 setting up topics.

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, multilevel topic presentation was developed to improve user foraging on big channels. Other additions such as the source recommender were prioritized when they became the biggest obstacles to user satisfaction.

As we learned about lean startup practices, we 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.

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 100,000 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 word indexes for articles when they were first collected. The classifier could then match articles to Boolean queries without rereading their contents.

We switched to a NoSQL database for article contents to support the millions of articles that the system now held. We periodically reworked slow queries and found more ways to precompute 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 more than 600 users and was able to run for several months without crashing. Videos of the Kiffets in operation are available on YouTube.

Concluding Remarks

Kiffets follows the knowledge is power logic of earlier AI systems in that it depends on the expertise of its curators. The system acquires curator expertise using a machine-learning approach where curators 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. It creates fresh, organized channels of information for readers every day.

The news business is 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), and how it is paid for (subscription services, pay walls, and advertising). This project opened a further dimension of change: how abundant news can be socially curated.

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

AAAI Returns to the Pacific Northwest for AAAI-13!

Please mark your calendars now for the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI-13) and the Twenty-Fifth Innovative Applications of Artificial Intelligence Conference (IAAI-13), which will be held in the Greater Seattle area, July 14-18, at the beautiful new Hyatt Regency Conference Center in Bellevue, Washington. Exciting plans are underway to coordinate with local University of Washington, Microsoft, and other members to make this a memorable event! Updates will be available at www.aaai.org/Conferences/AAAI/aaai13.php this summer.

and Markus Fromherz for their guidance and support. Thank you to Jim Pitkow for advice on lean management and 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: Hyperion.

Arthur, C. 2010. Eric Schmidt Talks about Threats to Google, Paywalls, and the Future. *Guardian.* London: Guardian News and Media Limited (online at www.guardian.co.uk/media/2010/jul/02/activate-eric-schmidt-google).

Burke, R.; Felfernig, A.; and Göker, M. H. 2011. Recommender Systems: An Overview. *AI Magazine* 32(3): 13–18.

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.

Dong, L.; Smith, R. G.; and Buchanan, B. G. 2011. NewsFinder: Automating an Artificial Intelligence News Service. *AI Magazine* 33(2).

Jones, K. S., and Willett, P. 1997. *Readings in Information Retrieval.* San Francisco: 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 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: Oxford University Press.

Ries, E. 2011. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses.* New York: Crown Publishing Group.

Salton, G., and Buckley, C. 1988. Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management* 24(5): 513–523.

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. 2011. We Digital Sensemakers. In *Switching Codes: Thinking Through Digital Technology in the Humanities and Arts,* ed. T. Bartscherer and R. Coover, 38–60. Chicago: The University of Chicago Press.

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

Mark Stefik is a research fellow at PARC. He is a fellow in the Association for the Advancement of Artificial Intelligence and also in the American Association for the Advancement of Science. He is currently engaged in two projects involving smart infrastructure as digital nervous systems: urban parking services for smart cities and digital nurse assistants for smart health care. Stefik has published five books, most recently *Breakthrough* in 2006 by the MIT Press. Stefik's website is www.markstefik.com.

Lance Good is a software engineer at Google where he works on Google shopping. He worked at PARC on Kiffets and several sensemaking systems. His previous work focused on using zoomable user interfaces for presentations, sensemaking, and television navigation. Good's website is goodle.org.