Discovery News: A Generic Framework for Financial News Recommendation

Chong Wang, Lisa Kim, Grace Bang, Himani Singh, Russell Kociuba, Steven Pomerville, Xiaomo Liu*
S&P Global Ratings
55 Water Street, New York, NY 10004
{chong.wang, lisa.kim, grace.bang, himani.singh, russell.kociuba, steven.pomerville, xiaomo.liu}@spglobal.com

Abstract

In the financial services industry, it is crucial for analysts to constantly monitor and stay informed on the latest developments of their portfolio of companies. This ensures that analysts are up-to-date in their analysis and provide highly credible and timely insights. Currently, analysts receive news alerts through manually created news alert subscriptions that are often noisy and difficult to manage. The manual review process is time-consuming and error-prone. We demonstrate Discovery News, a framework for an automated news recommender system for financial analysis at S&P's Global Ratings. This system includes the automated ingestion, relevancy, clustering, and ranking of news. The proposed framework is adaptable to any form of input news data and can seamlessly integrate with other data used for analysis like financial data.

Introduction

Analysts in the financial services industry ingest news from dozens of sources every day. It is critical for them to stay up-to-date on the latest events as at any moment, key material events could occur that impact their analysis on a given company. For example, a company could announce an acquisition or a disruption to their supply chain that would materially impact its business operations. By staying informed on these events in a timely manner, analysts can promptly respond to these changing circumstances and potentially minimize any impact of the company's action on their investment holdings or analysis.

Currently, for each company they analyze, the analysts manually set up news alert subscriptions from a fixed set of sources like Google or Bloomberg based on keywords of the company's name. Thus, the news alerts analysts receive contain much noise. Also, a company with a name like "Apple" could result in new alerts about apple orchards or apple juice. Or alternatively, if there are critical articles discussing changes in regulations of the smartphone industry but there are no explicit mentions of company names, this may not be picked up through the keyword generated manual news alert. The lack of comprehensive coverage means the attention to certain critical news events can be delayed or missed altogether. Additionally, when a breaking news event occurs, dozens of media outlets report on the same occurrence which result in multiple alerts on the same event. Manual review of relevant and important news events is time consuming and error-prone.

At S&P Global Ratings, identifying the relevance of the news event to the analytical assessment of a company is a complex process due to the nuanced definition of a credit rating. The credit rating of a company is a 1-to-2 year forward looking assessment of the company's performance and thus, material news events should impact the future performance of the company in a sustained and meaningful way. This impacts the approach required to assess the relevance of the news events to the end user. For example, certain topics like a management turnover could be breaking news but may not be impactful to the company's rating and thus not highly relevant for analysts. Additionally, for example, if Google engages in an activity in partnership with Walmart, if the activity were to primarily impact Google and only tangentially impact Walmart, the news event would not be as relevant for analysts covering Walmart.

Therefore, it is important to build a news recommender system for financial analysis, which can suggest important news articles that are relevant to financial analysts' tasks. To be applied in large financial companies, such system must be 1) effective so that analysts will not miss any important information, 2) efficient so that news can be timely updated, 3) scalable so that it can be used in large business environments, and 4) extendable so that changing business requirements can easily be met.

In this paper, we propose a framework for the automated ingestion, clustering, relevancy, and ranking of news events for financial analysts in the debt capital market. It is an ongoing project currently in the testing phase at S&P Global Ratings. More specifically, the method herein uses a combination of entity-based news tracking, clustering, relevancy, and ranking models to detect news events, filter out noise, cluster the same event into each of the respective clusters, assign relevance of news event to main entity, and rank each news event cluster based on the importance of the event. The

*corresponding author

ranked list of events are presented to the user in this curated manner which allows analysts to focus on key news relevant to their analysis and thus spend less time filtering through noise and reviewing redundant events. New business requirements can be easily added by defining new subscriptions with any customized entities. Processes within subscriptions can be fully paralleled.

Our contributions include: (1) a full-stack systematic methodology of information retrieval and relevancy to automatically distinguish financially material news events, and (2) a generic, scalable, and extendable news recommendation framework that can be easily adopted by most financial companies. To the best of our knowledge, this is the first such solution for recommending news for financial analysis. This news recommender is able to detect emerging risks to company's operations faster and more efficiently than current analyst methods.

Related Work

News recommendation systems have been extensively studied in the past ten years. Most studies contributing to news recommenders suggest news articles to users based on their interests with the ultimate goal of improving user satisfaction and boosting click rate.

Early news recommenders (IJntema et al. 2010) used external ontologies to find interesting news articles. However, related knowledge bases do not always exist and are highly expensive to build. In the recent years, many studies used collaborative filtering (CF) based methods (Abhinandan et al. 2007; Lu et al. 2015a) , which performed effectively in movie and product recommendations (Ackerman, Wang, and Chen 2018). Existing work (Liu, Dolan, and Pedersen 2010; Li et al. 2014; Lu et al. 2015b; Garcin et al. 2012; Li and Jiang 2016) also proposed CF-based news recommender systems. These news recommenders capture user preference by user behavior, such as click, like, and repost. However, CF-based methods are not feasible in task-oriented news recommender systems, especially in the finance industry because 1) interesting business events may be rare, such as bankruptcy and acquisition. Recommending news similar to the ones in analysts' history may bias the system towards frequent events. In this case, once occurred, rare events have minimal chances to be surfaced; 2) in financial firms, one analyst is assigned to monitor a given company or several analysts may analyze different aspects of a company. In this case, diverse "user interests" generates highly sparse user-news interaction, which discounts the performance.

Content-based recommendation is another commonly-used technique. In the application of news recommendation, most content-based methods, e.g., (Kompan and Bieliková 2010; Wen, Fang, and Guan 2012; Li et al. 2011), build user profiles based on past reading history and calculate the similarity between an incoming news and each user profile. Similar to the CF-base methods, in task-oriented news recommender systems, determining news relevancy based on the history of individual users may favor frequent business events than important but rare events.

Past studies also proposed popularity-based recommender systems (Jonnalagedda et al. 2016). They monitor news popularity from social media such as Twitter. However, task-oriented news systems should not be impacted by the popularity of news on social media.

Finally, few of existing studies present the full-stack of a news recommender system pipeline that can be easily transplanted and deployed in non-IT companies. Therefore, we present a task-oriented financial news recommender system that is being used and tested in a large financial company. We demonstrate the full-stack system from data storage to user interface. The system can be easily parallelized on standalone servers or cloud clusters. The system was also built to be extendable and scalable so that different functions within a company can customize it.

System Overview

We present the overall architecture of the S&P Global Ratings Discovery news recommender system, followed by detailed descriptions of each component.

System Architecture

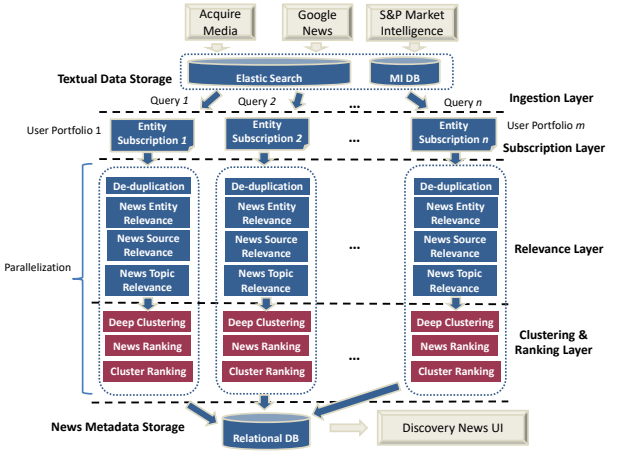

Figure 1: System Architecture

The system architecture of our Discovery news recommender system is shown in Figure 1. The system integrates news data from multiple different news providers. Without loss of generality, the presented framework can integrate data from any data providers.

Financial analysts have different companies in their portfolios and follow a variety of topics. Also, new business requirements are coming out over time. Thus, we propose a subscription-based pipeline where each entity (e.g., company name, location, or topic) is a subscription. All news articles talking about a company, e.g. Apple, are assigned into the subscription of Apple Inc. A subscription may be associated with multiple analysts as certain topics like merger and acquisitions are of interest to all analysts. Likewise, a news may be relevant to multiple entities (e.g., a news talking about Google and Facebook are expanding offices in New York City). Thus, a news article may also belong to multiple

subscriptions. By default, analysts receives news of the subscriptions in their portfolios but our system allows analysts to create and follow additional subscriptions. New business requirements can be met by creating new types of subscriptions and/or adding more subscriptions. Processes in each subscription are independent of each other. The pipeline can be fully parallelized by subscriptions.

Detailed Description

Data Collection: The system integrates heterogeneous data streams, i.e., Acquire Media News[1], Google News[2], and S&P's Market Intelligence (MI)[3]. Acquire Media is a third-party company that provides a news management and delivery solution for publishing and financial services. Acquire Media delivers about 60K daily news published by >10K global sources in XML format. Being stored on Amazon S3, each XML contains information on a news article, such as title, content, and source. Google News is a publicly available news aggregator developed by Alphabet Inc. Each entity of interest is set up as keywords in Google News as a feed. Google uses their own algorithms to collect news related the keywords. A crawler on the S&P end is scheduled to download news from all feeds. S&P Market Intelligence (MI) News is a third-party news provider that delivers curated news with an emphasis on analysis rather than reported news. The news distributed through the platform focuses on offering insights to end users.

Data Storage: Data collected from the above sources is stored into a NoSQL database, Elasticsearch [4], as the dataset is large, dynamic, and requires fast retrieval. The parsed data is directly inserted to Elasticsearch which automatically detects and adds new fields when required. The system uses a cluster of nodes for storage, different compound queries to prepare the data for the models and visualizations via Kibana to detect anomalies and enhance the models as needed.

The system uses a relational database to store the output of the clustering algorithm and maintain the atomicity of operations on the database and to comply with integrity requirements both in data type and compatibility for the services layer. The relational database stores the relationship between clusters, news stories, and user subscriptions to create data objects for the UI.

Parallelization: After news articles are distributed into subscriptions, all subsequent processes are conducted within each subscriptions. We accelerate the pipeline by adopting multi-processing to process subscriptions simultaneously: Subscriptions are assigned and processed on different CPUs. Since subscriptions may have different numbers of news, we use a greedy algorithm (shown in Algorithm 1) to guarantee that the CPUs get balanced workload. Note, the same parallelization design can be applied on Hadoop clusters due to the independence among subscriptions.

De-duplication: We then de-duplicate news in each subscription as: 1) some news sources copy and re-post news from the original source (especially well-known sources) with little changes. Thus, identifying duplicate news helps remove noise, improve the cluster importance estimation, and speed up the whole pipeline. 2) A news source may produce updated versions of news with minor changes. Thus, de-duplication module can reduce these news by detecting and only keeping the latest one.

Data: Subscriptions $S = \{s_1, s_2, ...s_n\}$; $|s_i|$ is denoted as the number of news in s_i
Result: A minimum heap that contains subscriptions allocated to each CPU
minimum Heap $H \leftarrow \emptyset$;
$j \leftarrow 0$;
while $j <$ *#CPUs* **do**
  $S'_j \leftarrow \emptyset$ // store allocated subscriptions;
  push$(H, S'_j, [0, j])$ // 0 is #news m in jth CPU;
  $j \leftarrow j + 1$;
end
$i \leftarrow 0$;
while $i < n$ **do**
  $m, S'_j, j \leftarrow$ pop(H) // jth CPU with the least #news;
  push$(H, [m + |s_i|, S'_j \cup \{s_i\}, j])$;
  $i \leftarrow i + 1$;
end

Algorithm 1: Subscription Allocation Algorithm

Since re-posted news may be modified slightly (e.g., adding the re-posting news source name), simply comparing if news are the same causes low recall. To enable Near-duplicate detection, we propose a Support Vector Machine (SVM) fed by the Local Sensitive Hashing (LSH) scores. LSH has been widely used in Near-duplicate detection (Das et al. 2007). Given a pair of news, we calculate the LSH signatures of their titles, descriptions, and contents. The Levenshtein distances and token sort ratios [5] of the title pair, the description pair, and the content pair are computed based on the LSH signatures. These six distances are fed into a SVM model with binary labels (i.e., whether the pair is duplicate). The SVM is trained over a manually-labeled training data by learning the parameters for these six features. A cut-off threshold is learned based on a validation dataset. The final model is evaluated on test data.

News Entity Relevance: It is common that, although an entity (e.g., a company or a city) is mentioned in a news article, it is not the focus of the article. Thus, we apply the news entity model to identify the relevance of an article to an associated entity and remove irrelevant news.

We use a machine learning model to identify main entities. We first derive features from the title, description, and content of each article. We apply phrase matching for name detection on the title and use spaCy[6] for name detection on

[1]http://www.acquiremedia.com
[2]https://news.google.com
[3]https://www.spglobal.com/marketintelligence
[4]https://www.elastic.co

[5]For two strings X and Y, the score is obtained by splitting the two strings into tokens and then sorting the tokens. It is to avoid the case that "Saleforce acquires Tableau" and "Tableau was acquired by Saleforce" are misclassified to be different.
[6]https://spacy.io

the description and content. In the latter case, we leverage spaCy to depict entity, noun, and subject recognition in text with standard sentence structure.

Since the title, description, and content have different purposes, independent features are generated respectively. Examples of the features are the following: for a title, we identify whether a title starts with the entity name, whether an entity name shows up in a sequence of entity names, and the normalized value of the first location of a entity name. For description and content, we generate a normalized mention frequency of a entity name in the text. But to successfully achieve this, we had to address the issue that the entity alias name is mentioned in a news article instead of the full official name. Thus, exact matching of the name of a subscription often leads to low recall and introduces noise in feature. To overcome this challenge, we leverage n-grams, count the number of tokens that match with each n-gram of a entity name, and then weight the counts exponentially.

We attempt logistic regression and XGBoost (Chen and Guestrin 2016). The performance of models is evaluated based on the recall and precision score, but we put more weight on the recall as the false negative is more critical for our use-case. For example, while analysts would not be hugely affected by receiving some irrelevant news to their portfolio, missing some of the major events, such as a merger and acquisition happening with a company that the analysts are analyzing will have a huge impact on their analysis.

News Source Relevance: News published by established news sources (i.e., publishers) tend to be more credible and accurate. They are more influential to the financial market and investors. Thus, filtering news by news sources is necessary in order to further refine the news candidate sets.

We use Alexa Rank[7] as a standard to measure the influence of news sources. Alexa rank reveals how influential a website is relative to all other sites based on a combination of a site's estimated traffic and visitor engagement.

We first build a mapping from news sources to domains: We group news articles by news sources and extract domains from the URLs. We keep all news domains in case the news articles published by a news source has multiple domains. The output of this step is a hash table where keys are news source names and values are the lists of possible candidate domains in descending order. Then, for each news source, we sequentially look up the list of domains in the Alexa Rank database by frequency until a match is made. News published by the bottom 25% ranked news sources and unranked news sources are filtered out. To ensure the system validity to the financial analysis process, we also develop a white-list of news sources provided by analysts.

News Topic Relevance: Analysts are keen on receiving news that could impact their analysis. Hence, we build a set of models to identify or score news topic relevance.

There are two categories of irrelevant news that can sift through our Elasticsearch queries: stock-related and patent-related news. Although these news articles mention entities (e.g., company name, geo-locations, or natural disasters), the events in the articles do not impact rating activities. Most news on stocks discuss company stock price movements, which do not impact the company analysis. Only stock news about the initial public offering (IPO) of a company or a large capital raise can impact the company analysis. Likewise, patent filings typically do not impact the analysis. However, lawsuits regarding patents infringement could affect the future performance of the company's products and consequently, income. Since both IPOs and patent lawsuits events have limited set of keywords, we use keyword-based classifiers to filter out irrelevant stock and patent news.

Aside from those explicit events, most events that can potentially impact rating activities are implicit and mostly determined by analysts' knowledge and interpretation of the events. We propose to build machine learning models to predict topic relevance. To provide training data, in the testing phase, analysts at S&P can choose to give us feedback for clusters, i.e, what degrees of relevancy a cluster has and why. As an ongoing project, we will build a deep neural network with softmax output to predict the user relevance feedback.

Clustering: The purposes of clustering news are 1) to signal the event importance as big news events may be reported by many news sources. Thus, the size of a news cluster can indicate the event noteworthiness (Liu et al. 2017), and 2) to improve the user experience. Grouping news on the same events improves user browsing experience as analysts do not have to manually skip over similar news in the application.

Traditional document clustering methods are mainly word/phrase-based (Watrous-deVersterre, Wang, and Song 2012) or topic-based algorithms (Wang and Blei 2011). However, the performance of keyword and named entity extraction is still unsatisfactory on open-domain documents.

We adopt deep learning methods that considers all words in a document without explicitly figuring out keywords or named entities. In particular, we adopt a state-of-the-art model, ELMo (Peters et al. 2018), to compute lower-dimensional representation for each news. ELMo first learns contextualized word representation by pre-training a language model in an unsupervised way. Unlike widely-used word embeddings (Pennington, Socher, and Manning 2014), ELMo word representations are functions of the entire input sentence (i.e., contextualized word representation). These word representations are computed on top of multi-layer bidirectional language models (biLMs) with character convolutions, as a linear function of the internal network states.

Figure 2 illustrates the structure of our ELMo network. At the input layer, ELMo first converts each token to an appropriate representation using character embeddings which is then run through a convolutional layer using some number of filters, followed by a max-pool layer. This representation is passed through a 2-layer highway network before being provided as the input to the biLMs layer. We can use character convolutions to learn morphological features and form a valid representation for out-of-vocabulary words. The biLMs learn embeddings for each word from different directions. To combine these embeddings into one, a weighted average are applied based on softmax normalization. To get a sentence representation, a fixed mean-pooling is applied over all contextualized word representations.

[7]https://www.alexa.com

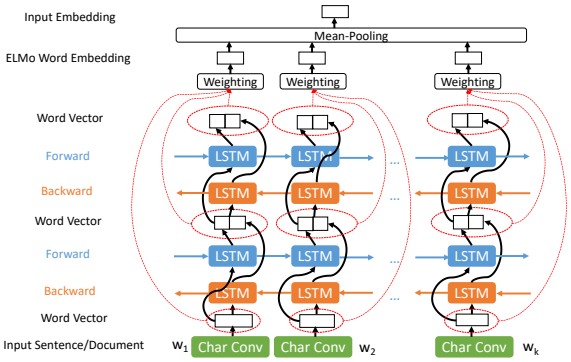

Figure 2: ELMo Structure

The ELMo news representations can be fed into any clustering algorithm, such as K-means and hierarchical clustering. In this project, we use a simple pseudo on-line clustering algorithm as shown in Algorithm 2. Generally, we first sort the news in chronological order. We also consider historical news clusters formed in the the previous day, i.e., t is set to one day. For each news received in the current cycle (e.g., today), we assign it to the nearest existing cluster. If the nearest is not found or there is no existing cluster, a singleton cluster is initialized by the news. We cluster the incoming news iteratively in chronological order, mimicking the way we received them. Finally, we clean up the clusters that have not been updated for one day.

Data: News received in the new cycle $D = \{d_1, d_2, ..., d_n\}$; Historical news clusters computed at the end of the previous cycle $C' = \{c'_1, c'_2, ..., c'_m\}$; A maximum similarity threshold λ; A time window t
Result: News clusters
sort D in chronological order by publication time;
$C \leftarrow C'$ // store current clusters;
for *each* d_i *in* D **do**
  if $|C| > 0$ **then**
    compute the pairwise cosine similarities S for d_i and the centroid of each c_j in C;
    if $min(S) < \lambda$ **then**
      add d_i to the nearest cluster in D;
      continue;
    end
  end
  create a singleton cluster for d_i and add to C;
end
Discard clusters that are not added any news within t;

Algorithm 2: Our Simple Pseudo-online Clustering

Ranking: To ensure that important news events do not get buried in the UI, we rank both clusters and the news in each cluster. Our goal is to rank important clusters higher (i.e., cluster-level ranking) and show the most representative news for each cluster (i.e., news-level ranking). The representative news of a cluster is the top-ranked news in the cluster.

For news-level ranking, we rank news within each cluster by two factors sequentially: 1) Publication date: It ensures that analysts can see newly-added news in those historical clusters, and 2) News ranking score: It is the weighted sum of the entity relevance score and news source relevance, and the weight of the relevance scores are set empirically based on analysts' feedback.

For cluster-level ranking, we rank clusters by three factors sequentially: 1) Update date: The updated date is a good indicator of the event recentness. It ensures analysts do not miss new events; 2) Clustering ranking score: It is computed by taking the maximum news ranking score of all the news within the cluster and summing the weighted cluster size. 3) Cluster size: More important clusters tend to have large sizes because more news sources may cover the news events.

User Interface (UI): Figure 3 is a screenshot of our UI. By default, the representative news of all clusters are presented with titles, entity names, descriptions, and publication dates. By clicking "View X More", analysts can unfold the rest of news in that cluster. On the right side, analysts can mark the whole cluster as read by clicking the check icon. In the current testing phase, analysts can also provide feedback on the cluster relevancy by clicking the dialog icon.

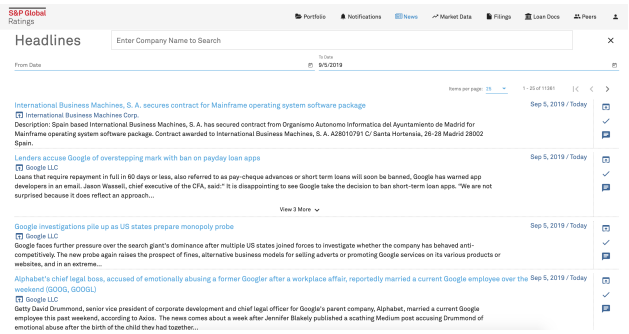

Figure 3: User Interface

Implementation & Evaluation

Implementation: The news recommender is running on an Amazon AWS server. In the current test phase, hundreds of financial analysts at S&P Global Ratings are using the system in their daily job. They contribute the relevance feedback on the UI for the news topic relevance model. As an offline process, the pipeline currently runs every morning from ingestion to news result generation. Results are stored in a relational DB. When an analyst logs on, UI will bring up and rank all news in the analyst's portfolio. In the future, the pipeline will be kicked off more frequently, e.g., every two hours, so that the news can be more timely updated.

News Entity Relevance: Based on the recall and precision scores, a fine-tuned XGBoost classifier is chosen as the final model. Table 1 illustrates the evaluation result of Logistic Regression, a baseline model, and XGBoost, the final model. While Logistic Regression has the recall score 1, the analysis on the results shows the reason is that the Logistic Regression model predicts all the news articles as 1, which

leads to the low precision score. The model underperforms when it cannot identify the entity name in the news article at all due to the use of an abbreviation or a pseudo-name. Some examples are Fidelity National Information Services Inc. and HP. We find that, in the misclassified articles, Fidelity National Information Services Inc. is referred to as FIS, and HP is called Hewlett-Packard.

Table 1: Evaluation of News Entity Relevance

	Logistic Regression	**XGBoost**
Recall	1.000	0.848
Precision	0.671	0.884
F1	0.803	0.866

Clustering: We also evaluate our clustering in terms of different news representations. The ELMo news representation is compared with 1) name entity-based (NER), in which spaCy and one-hot encoding are used to extract and model named entities, and 2) latent dirichlet allocation (LDA)-based, in which we use a LDA model to compute low-dimensional news representation. The clustering is done on a labeled news collection of 3 days. The performance is evaluated by adjusted rand score and adjusted mutual information score, two commonly-used metrics for clustering evaluation. The result show that the ELMo representation significantly outperforms the others.

Table 2: Evaluation of Clustering

	NER	**LDA**	**ELMo**
Adjusted Rand Score	0.0293	0.2843	0.3294
Adjusted MI Score	0.1126	0.2757	0.3268

Runtime: On average, the system takes about 15 minutes to process 1500 subscriptions. The entire process takes less than 45 minutes for nearly 5000 subscriptions.

Conclusions

The Discovery news recommender system offers several advantages that allow financial analysts to work more effectively and efficiently. In this paper, we present a full stack financial news recommender system from data storage to user interface that can be easily deployed at any financial company. With a subscription-based pipeline, the system can be conveniently parallelized and extended for new business monitoring requirements. By using a series of clustering and relevancy models, the system is able to recommend news events that have a direct relevancy to the final outcome of analysis, e.g., credit rating, at a financial firm.

Future work include training a relevancy model for credit rating based on the feedback received on UI and adding more types of subscriptions.

References

Abhinandan; Datar, M.; Garg, A.; and Rajaram, S. 2007. Google news personalization: scalable online collaborative filtering. In *WWW '07*.

Ackerman, B.; Wang, C.; and Chen, Y. 2018. A session-specific opportunity cost model for rank-oriented recommendation. *JASIS&T* 69(10):1259–1270.

Chen, T., and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *KDD '16*, 785–794. ACM.

Das, A. S.; Datar, M.; Garg, A.; and Rajaram, S. 2007. Google news personalization: scalable online collaborative filtering. In *WWW '07*, 271–280. ACM.

Garcin, F.; Zhou, K.; Faltings, B.; and Schickel, V. 2012. Personalized news recommendation based on collaborative filtering. In *WI-IAT '12*, 437–441.

IJntema, W.; Goossen, F.; Frasincar, F.; and Hogenboom, F. 2010. Ontology-based news recommendation. In *EDBT/ICDT Workshops '10*, 16. ACM.

Jonnalagedda, N.; Gauch, S.; Labille, K.; and Alfarhood, S. 2016. Incorporating popularity in a personalized news recommender system. *PeerJ Computer Science* 2:e63.

Kompan, M., and Bieliková, M. 2010. Content-based news recommendation. In *EC-Web '10*, 61–72.

Li, C., and Jiang, Z. 2016. A hybrid news recommendation algorithm based on user's browsing path. *ICIS '16* 1–4.

Li, L.; Wang, D.-D.; Zhu, S.-Z.; and Li, T. 2011. Personalized news recommendation: A review and an experimental investigation. *J. Comput. Sci. Technol.* 26(5):754–766.

Li, L.; Zheng, L.; Yang, F.; and Li, T. 2014. Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications* 41(7):3168–3177.

Liu, X.; Nourbakhsh, A.; Li, Q.; Shah, S.; Martin, R.; and Duprey, J. 2017. Reuters tracer: Toward automated news production using large scale social media data. In *IEEE Big Data '17*, 1483–1493. IEEE.

Liu, J.; Dolan, P.; and Pedersen, E. R. 2010. Personalized news recommendation based on click behavior. In *IUI '10*, 31–40.

Lu, Z.; Dou, Z.; Lian, J.; Xie, X.; and Yang, Q. 2015a. Content-based collaborative filtering for news topic recommendation. In *AAAI '15*, 217–223.

Lu, Z.; Dou, Z.; Lian, J.; Xie, X.; and Yang, Q. 2015b. Content-based collaborative filtering for news topic recommendation. In *AAAI '15*.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *EMNLP'14*, 1532–1543.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Wang, C., and Blei, D. M. 2011. Collaborative topic modeling for recommend scientific articles. In *KDD'11*, 448–456.

Watrous-deVersterre, L.; Wang, C.; and Song, M. 2012. Concept chaining utilizing meronyms in text characterization. In *JCDL '12*, 241–248.

Wen, H.; Fang, L.; and Guan, L. 2012. A hybrid approach for personalized recommendation of news on the web. *Expert Syst. Appl.* 39(5):5806–5814.