A Personalized and Context-Aware News Offer for Mobile Devices

Toon De Pessemier, Kris Vanhecke, and Luc Martens

iMinds - Ghent University, Dept. of Information Technology,
G. Crommenlaan 8 / 201, 9050 Ghent, Belgium
{toon.depessemier,kris.vanhecke,luc.martens}@intec.ugent.be

Abstract For classical domains, such as movies, recommender systems have proven their usefulness. But recommending news is more challenging due to the short life span of news content and the demand for up-to-date recommendations. This paper presents a news recommendation service with a content-based algorithm that uses features of a search engine for content processing and indexing, and a collaborative filtering algorithm for serendipity. The extension towards a context-aware algorithm is made to assess the information value of context in a mobile environment through a user study. Analyzing interaction behavior and feedback of users on three recommendation approaches shows that interaction with the content is crucial input for user modeling. Context-aware recommendations using time and device type as context data outperform traditional recommendations with an accuracy gain dependent on the contextual situation. These findings demonstrate that the user experience of news services can be improved by a personalized context-aware news offer.

Keywords: Recommender system, Context-aware, Real-time, Mobile, News, User evaluation

1 Introduction

Recommender systems are software tools and techniques providing suggestions for items to be of interest to a user such as videos, songs, or news articles. The consumption of these audiovisual media and the accessing of information always happen in a certain context [31], i.e. conditions or circumstances that significantly affect the decision behavior. This gave rise to the development of context-aware recommender systems (CARS), which take this contextual information into account when providing recommendations.

For various application domains, the user context has gained an increased interest from researchers [3]. For context-aware music recommendations for example, the user's emotions can be used as input by using support vector machines as emotional state transition classifier [16]. In the application domain of tourism for example, various applications use the current location or activity of the user to personalize and adapt their content offer to the current user needs [30,12]. Personal recommendations for points of interest can be provided based on the user's proximity to the venue [20].

In the domain of audiovisual media, more specifically news content, the influence of context on the consumption behavior and personal preferences is less obvious. However, research [42] has shown that the situation of the user (location, activity, time), as well as the device and network capabilities are important contextual parameters for context-aware media recommendations on smartphones.

The growth of the digital news industry and especially the development of mobile products is booming. Mobile has become, especially amongst younger media consumers, the first gateway to most online news brands. In a recent survey [29], conducted in 10 countries with high Internet penetration, one-fifth of the users now claim that their mobile phone is the primary access point for news. Despite this shift of news consumption to the mobile platform, the study of Weiss [40] highlights that a gap exists between what news consumers, particularly young adults, are doing and using on their smartphones and what news organizations are able to provide. In most cases, news organizations disregard contextual data or they are only using geo-location features in their mobile apps for traffic and weather; they do not anticipate the high use of location-based services by smartphone consumers.

In contrast to more traditional content domains of research on recommender systems, such as movies or books, news content are typically transient items. They are characterized by a short life span and quickly lose their information value over time. News items should therefore be recommended as soon as they are available in order to minimize delay between production and consumption of the content. For instance, a preview of a sports game has lost any information value after the game. Especially for online news, fast delivering and recommending of content is of utmost importance.

For content with a short life span, and for news content in particular, collaborative filtering (CF) systems have difficulties to generate recommendations because of the new item problem (cfr. cold start problem) [11]. CF requires a critical amount of consumptions (explicit or implicit feedback) before an item can be recommended. Once enough consumption data is available, the information value of the content might be degraded, making recommendations for the content useless. Therefore, content-based or hybrid approaches are considered as more suitable for news recommendation.

2 Related Work

In the domain of digital news services, various initiatives to personalize the offered news content have been proposed. One of the first recommender systems for personalizing news content was GroupLens [21]. GroupLens used collaborative filtering to generate recommendations for Usenet news and was evaluated by a public trial with users from over a dozen newsgroups. This research identified some important challenges involved in creating a news recommender system.

Another digital news service is SCENE [23]. It stands for a SCalable two-stage pErsonalized News rEcommendation system. The system considers characteristics such as news content, access patterns, named entities, popularity, and recency of news items when generating recommendations. The proposed news selection mechanism demonstrates the importance of a good balance between user interests, the novelty, and diversity of the recommendations.

The News@hand system [8] is a news recommender which applies semantic-based technologies to describe and relate news contents and user preferences in order to produce enhanced recommendations. This news system ensures multi-media source applicability and multi-domain portability. The resultant recommendations can be adapted to the current context of interest, thereby emphasizing the importance of contextualization in the domain of news. However, context is not the main focus of this study and the influence of context on the consumption behavior is not investigated.

The News Recommender Systems Challenge [32] focused on providing live recommendations for readers of German news media articles. This challenge highlighted why news recommendations have not been as analyzed as some of the other domains such as movies, books, or music. Reasons for this include the lack of data sets as well as the lack of open systems to deploy algorithms in. In the challenge, the deployed recommenders for generating news recommendations are: Recent Recommender (based only on the recency of the articles), Lucene Recommender (a text retrieval system built on top of Apache Lucene), Category-based Recommender (using the article's category), User Filter (filters out the articles previously observed by the current user), and Combined Recommender (a stack or cascade of two or more of the above recommenders).

The usefulness of retrieval algorithms for content-based recommendations has been demonstrated with experiments using a large data set of news content [6]. Binary and graded evaluation were compared and graded evaluation showed to be intrinsically better for news recommendations. This study emphasizes the potential of combining content-based approaches with collaborative filtering into a hybrid recommender system for news.

Although the various initiatives emphasize the importance of a personalized news offer, most of them focus on the recommendation algorithms and ignore the contextual information that is coupled with the information request, the user, and the device. In this study, the focus is not on improving state of the art recommendation algorithms, but rather on investigating the influence of context on the consumption of news content by means of a large-scale user study.

In many cases, the research on CARS remains conceptual [3], where a certain method has been developed, but testing is limited to an offline evaluation or a short-term user test with only a handful of people, often students or colleagues who are not representative for the population. In contrast, this research investigates the role of context for news recommendations by means of a large-scale empirical study. Users could utilize a real news service[1] that offers content of four major Flemish news companies on their own mobile devices, in their everyday environment, where and when they wanted, i.e., in a living lab environment.

Living lab experiments are an extension towards more natural and realistic research test environments [14]. Living labs allow to evaluate research hypotheses by users representative for the target population who satisfy their information need in a real context. Since users are following their own agenda, laboratory biases on their behavior can be neglected [18]. Although less transparent and predefined, living lab experiments aim to provide more natural settings for studying users' behavior and their experience [10].

Especially for context-aware applications, in which the user's environment has an influence on the way the application works and/or on the offered content, a realistic setting is essential for a reliable evaluation. Therefore, this paper investigates the influence

[1] http://www.iminds.be/en/projects/2014/04/17/stream-store

of context and the benefit of context-aware recommendations for a real news service by means of a large-scale user panel, in a realistic environment, over a longer period of time. Since a user study can provide reliable explanations as to which recommendation method is the best, and why one method is better than the other [33], three alternative recommendation methods for the news service are compared through such a user study.

3 Search Engines and Recommender Systems

To generate recommendations, a content-based approach was chosen because of the availability of informative metadata about the content items, the sparsity of the data set, and the cold start problem associated with the start-up phase (Section 1). Content-based algorithms typically compare a representation of the user model with (the metadata of) the content, and deliver the best matching items as recommendations [24]. These algorithms often use relatively simple retrieval models such as keyword matching or the Vector Space Model (VSM) with basic Term Frequency - Inverse Document Frequency (TF-IDF) weighting [25]. As such, the matching process of content and user model in a content-based algorithm shows many resemblances with the content retrieval process of a search engine.

Before employing the VSM and TF-IDF weighting in a content-based algorithm, preprocessing of the content is often required. If the content consists of complete sentences, the text stream must be broken up into tokens: phrases, words, symbols or other meaningful elements. Tokens that belong together, e.g. United States of America or New York, deserve special attention, and can be handled by reasoning based on uppercase letters and n-gram models [7]. Before further processing of the content, the next operation is filtering out stop words, the most common words in a language that typically have a limited intrinsic value. Another important operation is stemming, the process for reducing inflected (or sometimes derived) words to their word stem, or root form. In our implementation, Snowball [28] is used, a powerful stemmer for the different languages. Again, a resemblance with content retrieval processes can be noticed, since these preprocessing operations are also performed during the indexing of web pages in search engines.

Based on these resemblances between the content recommendation and content retrieval problem, we opted to utilize a search engine as the component for processing and indexing the content in our news service. Utilizing a search engine to process and index the news content brings some additional advantages.

– *Short response time.* Search engines are strongly optimized to quickly identify and retrieve relevant content items. An inverted index [9] is used as a very efficient structuring of the content, enabling to handle massive amounts of documents.
– *Fast processing of new content.* New content items can be processed quickly by making additions to the index structure, thereby making these new content items available for recommendation almost immediately. In contrast, traditional collaborative filtering systems often require intensive calculations of similarities before a new item can be recommended.
– *Limited storage requirements.* The index structure of search engines is a very efficient storage way to retrieve documents.

4 Recommendation Architecture

Figure 1 shows the architecture and content flow of the news recommender system. The five different phases will be discussed in more detail in this section.

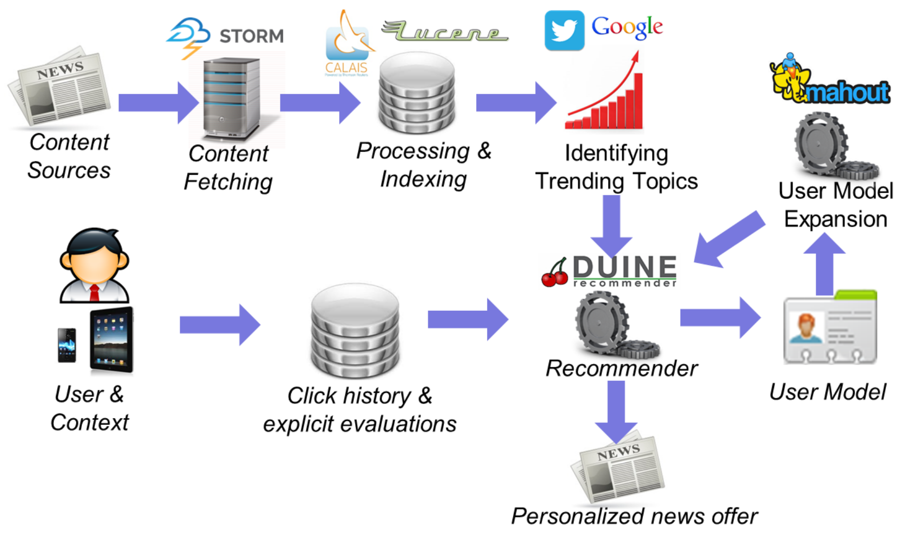

Figure 1. The architecture and content flow of the news recommender system.

4.1 Content Fetching

The first phase of the recommendation process consists of *fetching the news content* periodically from different sources. When new items are available, their content is fetched and processed. Many online news services provide their content through RSS-feeds. To parse these feeds, the Rome project [41] is used since this is a robust parser. Besides RSS-feeds, other sources, such as blogs, can also be incorporated into the system by using a specific content parser.

In order to keep track of the most recent news content, news sources are checked regularly for new content. Different news sources have a different publishing frequency, ranging from one news item per day, to multiple news items per minute. Therefore, we used a simple mechanism to adapt the frequency of checking for new content to the publishing frequency of the content source. For each content source, a dynamic timer is used to determine when to check for new content. After a timeout, the content is fetched. If new content is available, the content item is passed to the search engine and the timeout is reduced by half. If no new content is available, the timeout is doubled. This simple mechanism showed to be sufficient as a convergence method for the timeout parameter.

In order to process the stream of incoming news articles of different sources continuously, Apache Storm [4] was used. Storm enables the processing of large streams of data in real time. As opposed to batch processing, Storm handles the news articles as soon as these are available. To use Storm, a topology composed of ‘Spouts’ and ‘Bolts’ has to be built, which describes how messages flow into the system and how they have to be processed. A Spout is a source of data streams. A Bolt consumes any number of data streams, does some processing, and can emit new data streams. Storm can make duplicates of these components, and even distribute these duplicates over multiple machines, in order to process large amounts of data. As a result, Storm makes the system scalable and distributed.

Figure 2 visualizes the Storm topology of our implementation. The Spouts input data into the system as URLs of RSS-feeds, blogs, or social network accounts. Storm will distribute the work load over different Bolts of the first type, which fetch the data from the feeds. In case new articles are available in the feed, the URLs of these articles are passed to the Bolts of the second type. These Bolts fetch the article content and remove non-topical information, such as advertisements, by identifying specific HTML tags in the source code of the web page. Subsequently, the Bolts pass the article content to Bolts of the third type. The task of Bolts of the third type is to analyze the content and obtain information such as the title, date, category, etc. Next, the article content is passed to the fourth type of Bolts, which input the news articles into the processing engine. After inputting the content into the processing engine, statistical information about the article content is stored by the fifth and last type of Bolts. E.g., the frequency of occurrence of a term at a specific moment in time is used to determine if a news topic is trending and important (Section 4.3).

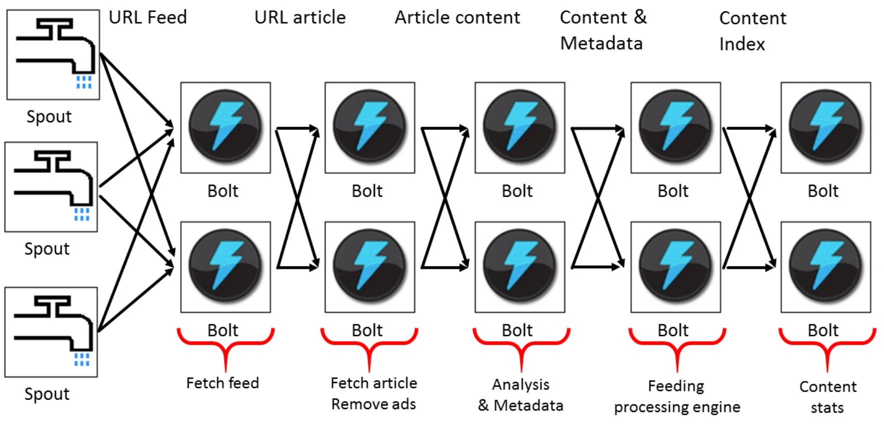

Figure 2. The Storm topology of our system.

4.2 Processing Engine

In the second phase, the content is *processed and indexed*. Given the equivalence between recommender engines and search engines [34], we opted to use a search engine for tasks such as indexing, retrieving items, and identifying n-grams.

Apache Lucene [36] was chosen as search engine, a Java library that is typically used for services handling large amounts of data and offering search functionalities. Since Lucene's performance, simplicity, and ease-of-use have been investigated in related work [17], this research does not focus on the characteristics of Lucene, but rather on the combination of search engine and recommender system.

As alternative search engines, we considered Apache Solr [38] and ElasticSearch [13]. Solr is a ready-to-use, open source search engine based on Lucene. In comparison with Lucene, Solr provides more specific features such as a REST webinterface to index and search for documents. However, the disadvantage of Solr is that some of the specialized functionality (that is needed in the recommendation algorithm) is hidden and not directly usable. Besides, the overhead of the webinterface of Solr introduced some delay in comparison with Lucene in our experiments. Similar to Solr, ElasticSearch hides some of Lucene's functionality by using a simple web interface. Specific information about the content items, such as the term frequencies or statistics about the complete index, are not directly accessible using ElasticSearch. Therefore, Lucene was chosen to provide the functionality of the search engine. In case the processing load for the Lucene index becomes an issue, distribution over different machines is possible by solutions such as Katta [19], thereby making it scalable.

News items in our system are characterized by eight different categories (national, international, culture, economy, lifestyle, politics, sports, and interesting facts), which are used to elicit the content preferences of the users through a questionnaire (Section 6.1). In addition, more detailed info of the news items is extracted through keywords and named entities, i.e. names of persons, organizations, locations, expressions of times, quantities, monetary values, etc. These keywords and named entities are not predefined but are extracted from the text of the article using OpenCalais [39]. OpenCalais is a Web service that automatically creates rich semantic metadata for the content. It analyzes the news article and finds the entities within it, but it returns the facts and events hidden within the text as well. This way, the news article is tagged and analyzed with the aim of checking whether it contains information what the user cares about.

4.3 Identifying Trending Topics

News events with a high impact (e.g., a huge natural disaster in a remote part of the world) have to be detected and considered as a recommendation, even if the topic does not completely match the user's interests. In the third phase of the recommendation process, these *trending topics* are identified based on their frequency of occurrence in the index of the search engine. If the current frequency of occurrence is significantly higher than the frequency of occurrence in the past, the topic is considered as trending.

Besides, trending topics are discovered by checking trends on Google's search queries [15]. Every hour, Google publishes a short list with trending searches. A special

Spout was implemented to fetch these trending topics hourly. Trending topics are used to create a query for the search engine, and the resulting news items are added to the user's recommendation list.

A final source of trending topics is Twitter. Research has shown that Twitter messages are a good reflection of topical news [27]. Therefore, another Spout was assigned specifically to query tweets regarding news topics using the Twitter API. Twitter accounts of specialized news services and newspapers were followed. The tweets originating from these accounts are focusing on recent news and characterized by a high quality. Retweets and Favorites give an indication of the popularity and impact of a tweet. Subsequently, Tweets are processed in the same manner as other news items by Bolts.

4.4 Generating Recommendations

In the fourth phase, personalized recommendations are generated. As content-based solution, the 'InterestLMS algorithm' of the Duine framework [35] was adopted. The InterestLMS algorithm is based on the VSM. It *builds a user model* by inferring personal preferences from the metadata describing the news items that are requested (implicit feedback) or evaluated (explicit feedback). As is common practice in the VSM [24], the user model is processed as a vector of terms (tags) together with a value specifying the user's interest in the term. These terms are words (or n-grams) in the article that are identified as relevant for the content (see Section 4.2).

Based on requests for reading news items (implicit feedback) or evaluations using the thumbs up/down functionality (explicit feedback), the user model is continuously *updated*. This feedback is transformed into a rating score. Thumbs up is mapped to the maximum rating, whereas thumbs down is the minimum rating. For implicit feedback, the amount of time spent on a news item (reading time for text or watching time for pictures and videos) is translated into a rating.

These item ratings are normalized (*normRating*) and then used to create or update the terms of interest in the user model that correspond to the metadata fields of the content item (equation 1).

$$newTerm_i = Aging * currentTerm_i + updateModerator * normRating * Nfactor \quad (1)$$

Here, the updateModerator is a constant that specifies the rate in which interests in topics are updated in the user model. The $Nfactor$ corrects for the number of terms ($\#terms$) that describe a content item: $Nfactor = 1/\#terms$.

In this update model, articles from the past are considered as less representative for the user's preferences than recent articles. Therefore, the *Aging* factor, a constant smaller than 1, decreases the contribution of terms of older news items.

Subsequently, the algorithm determines the news items that best match the user model reflecting the user's preferences. The interest terms in the user model are used to infer a *recommendation score* for each unseen news item based on the terms describing the item (equation 2).

$$recommendationscore = \sum_i currentTerm_i * weight(Term) \quad (2)$$

Here, the sum iterates over all terms identified in the unseen news item. $CurrentTerm_i$ stands for the preference value of term i in the user model. The weights specify the relative importance of different terms (e.g., categories are considered as more important than keywords). By ordering and filtering the unseen news items according to their recommendation score, a collection of suitable recommendations is generated based on the user's personal preferences for news content of different categories and characterized by different keywords.

4.5 User Model Expansion

As explained in the introduction, straightforward collaborative filtering is not usable for news recommendations because of the new item problem. Unfortunately, content-based recommendations are often characterized by a low serendipity; recommendations are too obvious. To introduce serendipity, a hybrid approach was taken by adding a collaborative filtering aspect to the content-based recommender. A traditional nearest neighbor approach was used to calculate similarities between user-user pairs. Instead of recommending the items that the neighbors have consumed (as in a collaborative filtering approach), our implementation will recommend terms that are prominent in the user models of neighboring users. These terms of the neighboring user models are used to expand the user model created by the InterestLMS algorithm, thereby making it more diverse. Subsequently, this expanded user model is used to generate content-based recommendations as described in Section 4.4.

By expanding the user model with terms that are significant in the model of the user's neighbors, user models are broadened and diversified with related terms. These expanded user models will produce more diverse recommendations covering a broad range of topics. Since the additional terms are originating from neighbors' user model, the added terms will probably be in the area of interest of the user. The collaborative filtering component is based on the implementation of Apache Mahout [37]. Mahout ensures the scalability of this component of the system. Moreover, the user model expansion is not a time-critical component, and is therefore implemented as a batch process running periodically. Content-based recommendations are based on the current version of the user model, and as soon as the model expansion is finished, the user model is updated. This ensures that real-time recommendations can be generated at all time.

5 Experimental Setup

The news service that was used in this experiment, aggregates content of different premium content providers: newspapers, magazines, but also content of television as short video clips. Figure 3 shows a screenshot of the user interface offering the content of different providers. The aggregated content offers users a more complete and varied overview of the news than traditional services do. To anticipate the abundance of news content and the associated choices that people have to make, the news service offers personalized recommendations.

During the experiment, the device type (smartphone or tablet) and the time of the day (morning, noon, daytime, or evening) are studied as contextual influences of the news consumption. The news service is accessible through a mobile application, which is available on Android and iOS for tablets as well as smartphones. As a result, the type of device that is used for consuming the news content is an interesting contextual factor.

Compared to the well-established application domains of recommender systems, such as movies or books, news items have a shorter lifespan and are frequently updated. Consequently, consulting the news on a daily basis, or even multiple times a day, can be interesting, which makes the time aspect another important contextual factor. The time is closely related to the location of the user, as was also witnessed during the analysis of users' interactions and consumption behavior. A frequently recurring pattern was as follows: in the morning, users are at home; during daytime, they are at work; and during the evening, they are again at home. Therefore, and to prevent over-specification (Section 6.3), the location is not adopted as a separate contextual factor.

For the evaluation of this service and its recommendations, 120 test users were recruited by an experienced panel manager from iMinds-iLab.o[2] (i.e. a research division with a strong expertise in living lab research and panel management). These test users, all interested in news and owning a smartphone and/or tablet, belong to the target group of an online news service. The test users could install the mobile application of the news service on their smartphone and/or tablet and freely use the service during the evaluation period of around 5 weeks. These test users were divided into three groups, each receiving a different type of recommendations, as explained in Section 6.

The test users' interactions with the service were logged to analyze their consumption behavior and to get insight in the actual use of the news service and their overall experience: 10 test users did not install the app, or did not use the news service during the evaluation period. They are excluded from the analysis, so that the number of actual participants was reduced to 110.

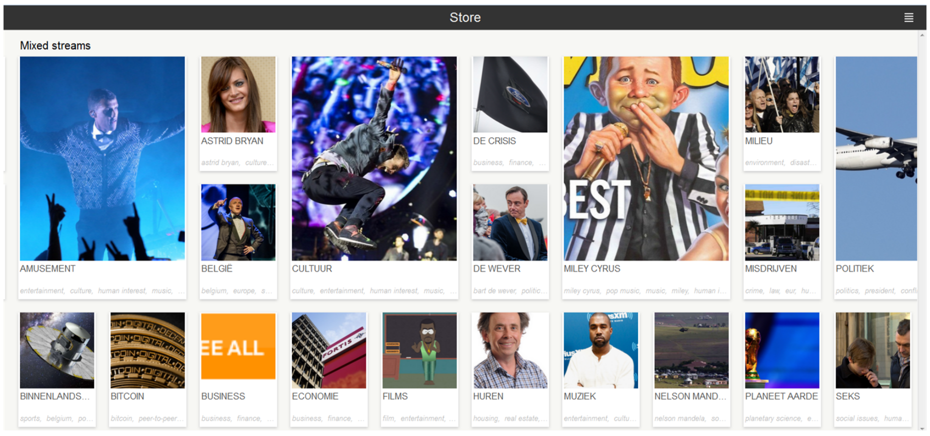

Figure 3. Screenshot of the user interface of the news service.

[2] http://www.openlivinglabs.eu/livinglab/iminds-ilabo

6 Three News Recommendation Approaches

The experiment takes three different approaches to recommend interesting news. Each user received only one type of recommendations during the whole evaluation period. To avoid any bias, test users were not informed about the existence of multiple types of recommendations.

6.1 Recommendations Based on Explicit Static Preferences

Before the actual experiment, test users were asked about their preferences for the eight different categories of news content (Section 4.2) through an online questionnaire. Users could specify their interests on a 5-point rating scale for each category and refine this score for different times of the day (morning, noon, daytime, and evening). The answers on this questionnaire constitute the user model that is used for generating news recommendations (Section 4.4). During the experiment, these preferences are considered static; user models are not updated based on explicit or implicit feedback on the content, and the recommender is not learning from the user's behavior.

6.2 Content-Based Recommendations

The content-based recommendations are not based on a prior questionnaire but use the implicit and explicit feedback users provide during the evaluation period. A request to read one of the recommended news items is considered as positive implicit feedback. Evaluating the news recommendation by means of the 'thumbs up' and 'thumbs down' icons in the user interface provides explicit feedback.

This feedback is gradually collected during the usage of the service. As a result, the user models of the content-based recommender are dynamic and constantly change as users interact with the news service. As the user is utilizing the news service and provides feedback, the recommender is learning the user's preferences.

For the content-based recommendations, contextual aspects are not taken into account. So, contextual data, such as the device type and the time of the day, are ignored during the creation of the user model and the calculation of the recommendations. In Section 4.4, more details about the recommendation algorithm are provided.

6.3 Context-Aware Content-Based Recommendations

Just like the content-based recommendations, the context-aware content-based recommendations are not using a prior questionnaire but are self-learning based on the explicit and implicit feedback users provide during the experiment. For this type of recommendations, the content-based algorithm is extended to take into account the context of the user. Before generating the recommendations, the user feedback is processed by a contextual pre-filter [2]. Contextual information is used to determine the relevance of the feedback and filter these data based on the current situation. For instance, if a user wants to read news during the evening, an *exact pre-filter* [3] selects only feedback gathered during the evening to calculate the recommendations. Therefore, the day is partitioned into four non-overlapping intervals: morning from 6:00 to 11:00, daytime from 11:00 to 12:00 and from 13:00 to 18:00, noon from 12:00 to 13:00 and evening/night from 18:00 to 6:00.

One major advantage of the contextual pre-filtering approach is that it allows deployment of any of the traditional recommendation techniques [1]. This makes it possible to use the same underlying algorithm (Section 4.4) for the context-aware content-based recommendations as for the content-based recommendations, which enables the comparison of both types of recommendations and to investigate the influence of contextual information.

Different pre-filtering techniques have proven their efficacy in literature [5]. They all have to cope with the problem of context over-specification: focusing on the exact context is often a too narrow limitation. An overly specified context may not have enough training examples for accurately estimating the user's interests. For example, if a user rarely utilizes a tablet during noon to read news articles, the exact context (noon + tablet) may not provide enough data (feedback from the user) for an accurate user model, which gives rise to the 'sparsity' problem. As a result, insufficient feedback is available for generating reliable recommendations [26].

An appropriate solution for context over-specification is to use a more general context specification by applying context generalization [3]. Since certain aspects of the overly specific context may be less significant, the data filtering can be made more general in order to retain more data after the filtering for calculating recommendations.

In this experiment, context generalization is applied in two phases in case of insufficient feedback data. In a first phase, the time frame is broadened. For instance, if recommendations are needed for a user who is reading news on a tablet during noon, the time restriction "noon" is dropped first. The data gathered in that specific context is supplemented with the user's feedback gathered on a tablet during other time periods. If the amount of feedback is still insufficient after this first generalization, the context is further generalized. In a second phase, the device type is broadened. More specifically, the user's feedback provided on a specific type of device (e.g., a tablet) is supplemented with the user's feedback provided on other device types (e.g., a smartphone). We opted to apply the generalization first on the time aspect of the context, and in a second phase on the device type, since many users are utilizing the service during different time periods but typically prefer one type of device per time period.

7 Results

7.1 Usage Patterns Throughout the Day

Figure 4 shows the amount of user interaction with the service (i.e., selecting a news item to view), aggregated over all users, for each hour of the day and per device type. A clear pattern in the consumption behavior is visible throughout the whole day. The close relation between time and location (Section 5) may have strengthened this pattern.

As expected, the amount of activity with the service is limited during the night. In the morning, users are very interested in the news, which is reflected in a peak in the service usage. During the day, the amount of consumptions varies slightly per hour with a slight increase around noon on tablet. During the evening, users spend more time reading news, which is revealed in Figure 4 by the increased user activity.

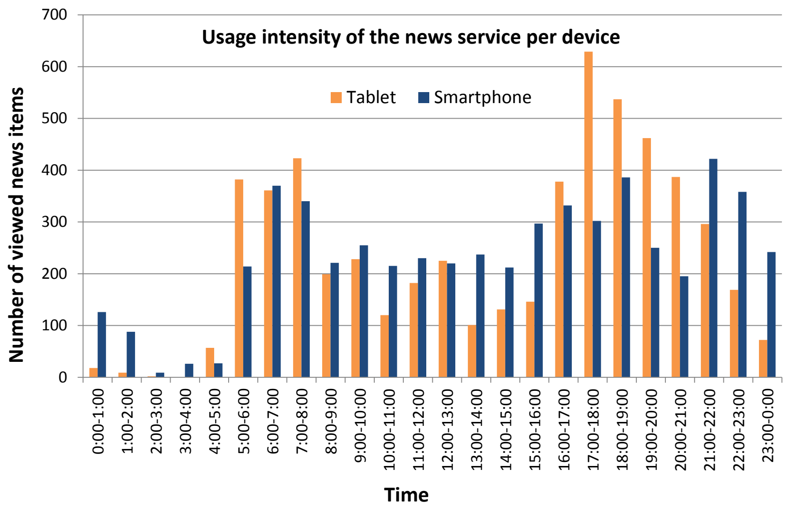

Figure 4. The amount of user interaction with the news service, aggregated over all users and partitioned according to the hour of the day.

Comparing both device types (smartphone vs. tablet) reveals that tablets are commonly used for consulting the news during the morning and evening. In contrast, smartphones are the primary device for consulting news during daytime. This analysis confirms the assumption that the usage of a news service and the amount of news content that is read, is strongly linked to the time of the day and the device type.

7.2 Three Recommendation Approaches Evaluated

To quantify the added value of a dynamic user model and contextual information, the users' interactions (implicit and explicit feedback) with each of the three types of recommendations are analyzed. Figure 5 gives an overview of the user feedback on the news service obtained during the evaluation period. The chart distinguishes implicit feedback, i.e. requesting to 'view' a news item, and explicit feedback, i.e., evaluating a news item by providing a 'thumbs up' or 'thumbs down' rating.

In Figure 5, this user feedback is aggregated over all users and partitioned by the type of recommendations that the users received. Since some test users dropped out just before the evaluation period, the different recommender types are not evaluated by the same number of test users. Table 1 shows the number of test users assigned to each type of recommendations, which has a direct influence on the total amount of feedback gathered for that type.

During the evaluation period, 2931 positive evaluations ('thumbs up') of a news recommendation were registered for all types of recommendations together. In contrast,

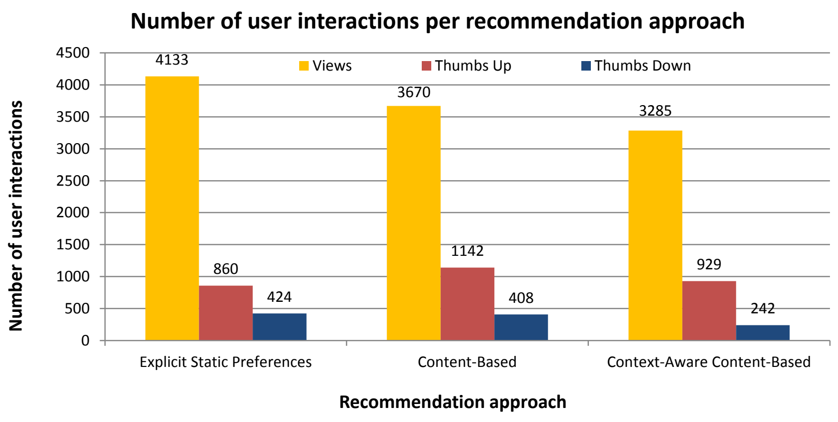

Figure 5. The amount of user interaction with the news service for each type of recommendations, partition according to the type of interaction.

Table 1. Comparison of the recommendation approaches

Recommendation Approach	Input Data	Number of Test Users	$\frac{\#Thumbs\,up}{\#(Thumbs\,up+down)}$
Explicit Static Preferences	Questionnaire	38	66.9%
Content-Based	Feedback	37	73.7%
Context-Aware Content-Based	Feedback + Context	35	79.3%

only 1074 times a negative evaluation (‘thumbs down’) was provided by the users. These aggregated values (‘thumbs up’ 73.1% - ‘thumbs down’ 26.9%) are an indication for the general satisfaction of the users with the news that they get recommended.

However, significant differences between the different types of recommendations can be witnessed. Compared to the recommendations that are based on the explicit static preferences of the users, less views (total number, but also number per user) are obtained for the content-based and context-aware content-based recommendations. This indicates that users get the interesting news more quickly using the more advanced algorithms, since they also spent more time per news item.

Comparing the different types of recommendations in terms of negative feedback (‘thumbs down’) demonstrates the added value of personal feedback and the context of the user for the recommender system. Recommendations based on explicit static preferences received 424 times ‘thumbs down’ from users who are not satisfied with the news content. The content-based recommender uses the implicit feedback (requests to view a news item) and explicit evaluations (‘thumbs up & down’) as personal feedback during the evaluation period. Compared to the recommendations based on explicit static preferences, less negative (408 times ‘thumbs down’) and more positive evaluations are provided for the content-based recommendations. The lowest number of negative evaluations (242 times ‘thumbs down’) was achieved with the context-aware content-based recommender, which suggests news items based on the personal feedback of the user and by taking into account the user's current context. A Wilcoxon rank-sum test showed these differences are significant ($p = 0.04 < 0.05$).

Table 1 shows the ratio of the number of positive evaluations (# thumbs up) and the number of explicit evaluations (# thumbs up + down) for the different types of recommendations. The results confirm the increase in accuracy by making the system dynamic (content-based recommendations) and taking into account the context (context-aware content-based recommendations). The improvement obtained by dynamic profiles is further investigated in Section 7.3, and Section 7.4 discusses the influence of context on the recommendation accuracy.

7.3 Accuracy Improvement through Dynamic Profiles

The accuracy increase obtained by making the user model dynamic is further investigated. Figure 6 shows the users' interests in news content of different categories. These interests can be obtained by explicitly asking the user through a questionnaire (as done in the recommendation approach based on explicit static preferences) or by deducing them from the actual user interactions with the service (as done in the content-based recommendation approach). These two methods are compared in Figure 6, which shows a clear difference. In general, users express a higher degree of interest in all news categories through the questionnaire compared to their actual interaction behavior with the news content. In other words, their expressed interests are only partly reflected in their selection of news content. This discrepancy might be due to time constraints of the user or due to the fact that the user can consult other information sources for news content.

More importantly is the difference in relative importance of the various news categories. Through the questionnaire, users express most interest in categories such as national news, international news and politics, whereas lifestyle is the least interesting category for them. In contrast, analyzing the actual user behavior results in opposite conclusions. Lifestyle is the most popular news category in terms of the number of views during the evaluation period. A possible explanation for this contrast is social desirability, i.e. the tendency of survey respondents to answer questions in a manner that will be viewed favorably by others.

These results show a significant difference between what users state that is interesting for them, and what users actually select and consume. As a result, making profiles dynamic by incorporating interaction behavior in the user model is of crucial importance to adjust the recommendations to the users' actual interests.

7.4 Accuracy Improvement through Context

To investigate the influence of context, the accuracy of the recommendations is investigated in different contextual situations. Table 2 shows the ratio of the number of positive evaluations (# thumbs up) and the number of explicit evaluations (# thumbs up + down) for different device types (smartphone and tablet). This tables compares the recommendations based on explicit static preferences and the context-aware content-based recommendations in terms of received feedback from the test users.

For the recommendations based on explicit static preferences, the results show a lower accuracy for recommendations made on tablet (62.9%) compared to smartphone

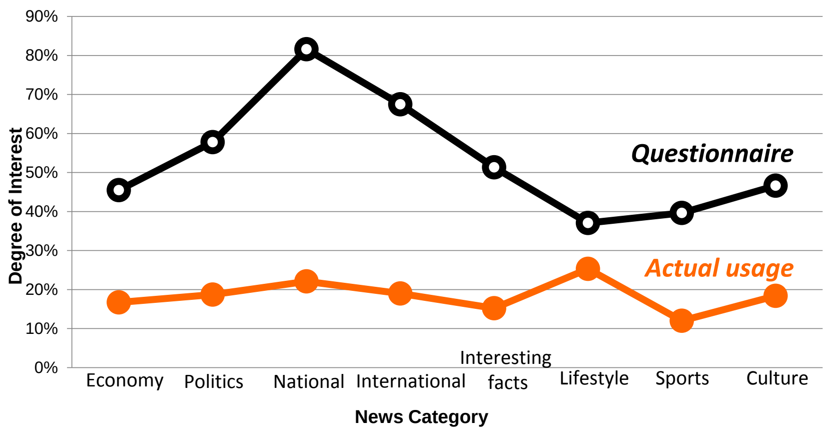

Figure 6. The users' interests per news category: users opinion (questionnaire) vs. actual usage (mobile app).

(73.5%). The accuracy of the context-aware content-based recommendations is higher in both context situations. But the accuracy difference between smartphone (79.6%) and tablet (75.2%) is reduced (from 10.6% to 4.4%) if the context is taken into account. This shows that the context-aware content-based recommender adjusts the news content to the contextual situation of the tablet. For example, news items that discuss specialized topics in large detail may be better suited for tablet devices than for smartphones. In addition, users may have specific preferences or habits regarding device type and content categories, such as reading sport news on smartphone and political news on tablet. The context-aware content-based recommender is able to automatically learn these preferences thereby recommending content adjusted to the device type.

Table 3 shows the ratio of the number of positive evaluations (# thumbs up) and the number of explicit evaluations (# thumbs up + down) at different time periods (morning, daytime, noon, evening). Again, the accuracy is compared for recommendation based on explicit static preferences and context-aware content-based recommendations.

For the recommendations based on explicit static preferences, the results show a rather stable accuracy over the different time periods. As shown in Section 7.1, the typical periods to consult the news are the morning and the evening. During noon and daytime, less time is spent on reading news articles. However, it is important to provide users an adjusted news offer during these time periods, such as an update of the morning news. Context-aware content-based recommendations take the time period into account, thereby achieving an accuracy gain of 1.6% and 8.9% for respectively morning and evening, i.e. the time periods most users consult the news. For the less typical time periods, the accuracy gain is much higher by exploiting the time-dependent needs of the users. The context-aware content-based algorithm improves the recommendation accuracy with 17.9% and 18.7% respectively during daytime and noon.

Table 2. Comparison of the recommendation results per device type

Recommendation Approach	Context	$\frac{\#Thumbs\,up}{\#(Thumbs\,up+down)}$
Explicit Static Preferences	Tablet	62.9%
Context-Aware Content-Based	Tablet	75.2%
Explicit Static Preferences	Smartphone	73.5%
Context-Aware Content-Based	Smartphone	79.6%

Table 3. Comparison of the recommendation results per time period

Recommendation Approach	Context	$\frac{\#Thumbs\,up}{\#(Thumbs\,up+down)}$
Explicit Static Preferences	Morning	65.4%
Context-Aware Content-Based	Morning	67.0%
Explicit Static Preferences	Daytime	65.4%
Context-Aware Content-Based	Daytime	83.3%
Explicit Static Preferences	Noon	70.3%
Context-Aware Content-Based	Noon	89.0%
Explicit Static Preferences	Evening	68.3%
Context-Aware Content-Based	Evening	77.2%

7.5 Discussion

While the context-aware content-based recommendations received the most positive evaluation in this experiment (highest ratio in Table 1), the gain in accuracy with respect to the static user model may become bigger over time.

In this experiment, the results are based on the evaluation period of approximately 5 weeks, and CARS require sufficient time to learn user preferences in different contextual situations. Because of the cold start problem (i.e. the issue that the system cannot draw any inferences for users or items about which it has not yet gathered sufficient information) and data fragmentation over the different contextual situations, we believe that there is still room for accuracy improvement of the context-aware content-based recommendations by gathering additional user feedback over a longer time period. The required number of ratings to overcome the cold start problem depends on various factors such as the algorithm parameters, the content domain, and the specific items that are rated. Studies have shown that, in general, more than 20 to 30 ratings are necessary for the system to recommend relevant items to the user [22]. In our user study, some of the users did not achieve enough ratings for each contextual situations. Additional data can help to learn patterns in the users' behavior and preference differences for various contextual situations, thereby further improving the accuracy of the context-aware content-based recommendations.

8 Conclusions

In this paper, a start-up news service offering personal recommendations is evaluated by an empirical user study. The typical characteristics of news content, such as the short-term life, and the limitations of a start-up service, such as a limited community of active users, make pure collaborative filtering techniques unusable. Therefore, the recommendation engine combines features of a content-based algorithm with a search engine and collaborative filtering using technologies such as Storm, Lucene, Duine, and Mahout. Storm enables the fast processing of large streams of news content. Lucene provides the functionality of a search engine and is used for processing and indexing the content. The Duine recommender framework is used to generate the content-based recommendations. The collaborative filter of Mahout is used to exchange terms of the user model among neighboring users. User models, as used in the content-based algorithm, are expanded with related terms interesting to read about. In the user experiment, three types of recommendations are tested: recommendations based on an explicit static user model, content-based recommendations using the actual user behavior but ignoring the context, and context-aware content-based recommendations incorporating user behavior as well as context.

The study aimed to assess the importance of context in the recommender of a real-life mobile news service by focusing on two contextual aspects: device type and time. The results confirm the added value of contextual information for personalized news recommendations by an increased recommendation accuracy. A more profound analysis showed that in specific contextual situations, a bigger accuracy gain can be obtained by using context-aware recommender systems, whereas in other situations the accuracy gain is limited. In this experiment, the context-aware algorithm obtained the best results on tablet devices and during time periods that are less typical for news consumption, such as during daytime and at noon. As future work, we consider to make a distinction between short-term interests and long-term interests of users. We also plan to focus more on entities mentioned in articles.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (June 2005)
2. Adomavicius, G., Tuzhilin, A.: Tutorial on context-aware recommender systems. In: Proceedings of the second ACM conference on Recommender Systems (RecSys '08) (2008)
3. Adomavicius, G., Tuzhilin, A.: Context-aware recommender systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 217–253. Springer US (2011), http://dx.doi.org/10.1007/978-0-387-85820-3_7
4. Apache Software Foundation: Apache storm (2015), available at http://storm.apache.org/
5. Baltrunas, L., Ricci, F.: Context-based splitting of item ratings in collaborative filtering. In: Proceedings of the Third ACM Conference on Recommender Systems. pp. 245–248. RecSys '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1639714.1639759
6. Bogers, T., van den Bosch, A.: Comparing and evaluating information retrieval algorithms for news recommendation. In: Proceedings of the 2007 ACM Conference on Recommender Systems. pp. 141–144. RecSys '07, ACM, New York, NY, USA (2007), http://doi.acm.org/10.1145/1297231.1297256

7. Brown, P.F., deSouza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Comput. Linguist. 18(4), 467–479 (Dec 1992), `http://dl.acm.org/citation.cfm?id=176313.176316`
8. Cantador, I., Bellogín, A., Castells, P.: News@hand: A semantic web approach to recommending news. In: Nejdl, W., Kay, J., Pu, P., Herder, E. (eds.) Adaptive Hypermedia and Adaptive Web-Based Systems, Lecture Notes in Computer Science, vol. 5149, pp. 279–283. Springer Berlin Heidelberg (2008), `http://dx.doi.org/10.1007/978-3-540-70987-9_34`
9. Cutting, D., Pedersen, J.: Optimization for dynamic inverted index maintenance. In: Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 405–411. SIGIR '90, ACM, New York, NY, USA (1990), `http://doi.acm.org/10.1145/96749.98245`
10. De Pessemier, T., De Moor, K., Joseph, W., De Marez, L., Martens, L.: Quantifying subjective quality evaluations for mobile video watching in a semi-living lab context. Broadcasting, IEEE Transactions on 58(4), 580–589 (Dec 2012)
11. De Pessemier, T., Coppens, S., Geebelen, K., Vleugels, C., Bannier, S., Mannens, E., Vanhecke, K., Martens, L.: Collaborative recommendations with content-based filters for cultural activities via a scalable event distribution platform. Multimedia Tools and Applications 58(1), 167–213 (2012), `http://dx.doi.org/10.1007/s11042-010-0715-8`
12. De Pessemier, T., Dooms, S., Martens, L.: Context-aware recommendations through context and activity recognition in a mobile environment. Multimedia Tools and Applications 72(3), 2925–2948 (2014), `http://dx.doi.org/10.1007/s11042-013-1582-x`
13. Elastic: Elasticsearch (2015), available at `https://www.elastic.co/`
14. Følstad, A.: Living labs for innovation and development of information and communication technology: A literature review. Electronic Journal of Organizational Virtualness 10, 99–131 (2008)
15. Google: Google Hourly Trends (2015), available at `http://www.google.com/trends/hottrends/atom/hourly`
16. Han, B.J., Rho, S., Jun, S., Hwang, E.: Music emotion classification and context-based music recommendation. Multimedia Tools Appl. 47(3), 433–460 (May 2010), `http://dx.doi.org/10.1007/s11042-009-0332-6`
17. Hatcher, E., Gospodnetic, O.: Lucene in action (in action series) (2004)
18. Hopfgartner, F., Kille, B., Lommatzsch, A., Plumbaum, T., Brodt, T., Heintz, T.: Benchmarking news recommendations in a living lab. In: Kanoulas, E., Lupu, M., Clough, P., Sanderson, M., Hall, M., Hanbury, A., Toms, E. (eds.) Information Access Evaluation. Multilinguality, Multimodality, and Interaction, Lecture Notes in Computer Science, vol. 8685, pp. 250–267. Springer International Publishing (2014), `http://dx.doi.org/10.1007/978-3-319-11382-1_21`
19. Katta: Lucune & more in the cloud (2015), available at `http://katta.sourceforge.net/`
20. Kenteris, M., Gavalas, D., Mpitziopoulos, A.: A mobile tourism recommender system. In: Proceedings of the The IEEE symposium on Computers and Communications. pp. 840–845. ISCC '10, IEEE Computer Society, Washington, DC, USA (2010)
21. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Grouplens: Applying collaborative filtering to usenet news. Commun. ACM 40(3), 77–87 (Mar 1997), `http://doi.acm.org/10.1145/245108.245126`
22. Lee, H., Kim, J., Park, S.: Understanding collaborative filtering parameters for personalized recommendations in e-commerce. Electronic Commerce Research 7(3-4), 293–314 (2007), `http://dx.doi.org/10.1007/s10660-007-9004-7`

23. Li, L., Wang, D., Li, T., Knox, D., Padmanabhan, B.: Scene: A scalable two-stage personalized news recommendation system. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 125–134. SIGIR '11, ACM, New York, NY, USA (2011), `http://doi.acm.org/10.1145/2009916.2009937`
24. Lops, P., de Gemmis, M., Semeraro, G.: Content-based recommender systems: State of the art and trends. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 73–105. Springer US (2011), `http://dx.doi.org/10.1007/978-0-387-85820-3_3`
25. Manning, C.D., Raghavan, P., Schütze, H., et al.: Introduction to information retrieval, vol. 1. Cambridge university press Cambridge (2008)
26. Papagelis, M., Plexousakis, D., Kutsuras, T.: Alleviating the sparsity problem of collaborative filtering using trust inferences. In: Herrmann, P., Issarny, V., Shiu, S. (eds.) Trust Management, Lecture Notes in Computer Science, vol. 3477, pp. 224–239. Springer Berlin Heidelberg (2005), `http://dx.doi.org/10.1007/11429760_16`
27. Phelan, O., McCarthy, K., Smyth, B.: Using twitter to recommend real-time topical news. In: Proceedings of the Third ACM Conference on Recommender Systems. pp. 385–388. RecSys '09, ACM, New York, NY, USA (2009), `http://doi.acm.org/10.1145/1639714.1639794`
28. Porter, M.F.: Snowball: A language for stemming algorithms (2001), available at `http://snowball.tartarus.org/`
29. Reuters Institute for the Study of Journalism: Digital News Report (2014), available at `http://www.digitalnewsreport.org/`
30. Ricci, F.: Mobile recommender systems. Information Technology & Tourism 12(3), 205–231 (2010)
31. Ricci, F.: Contextualizing recommendations. In: ACM RecSys Workshop on Context-Aware Recommender Systems (CARS '12), In conjunction with the 6th ACM Conference on Recommender Systems (RecSys '12). ACM (September 2012)
32. Said, A., Bellogín, A., de Vries, A.: News recommendation in the wild: Cwis recommendation algorithms in the nrs challenge. In: Proceedings of the 2013 International News Recommender Systems Workshop and Challenge. NRS. vol. 13 (2013)
33. Shani, G., Gunawardana, A.: Tutorial on application-oriented evaluation of recommendation systems. AI Communications 26(2), 225–236 (2013)
34. Shaphira, B., Rokach, L.: Recommender systems and search engines–two sides of the same coin? Slide Lecture `http://medlib.tau.ac.il/teldan-2010/bracha.ppt` (2012)
35. Telematica Instituut / Novay: Duine Framework (2009), available at `http://duineframework.org/`
36. The Apache Software Foundation: Apache Lucene (2015), available at `https://lucene.apache.org/`
37. The Apache Software Foundation: Apache Mahout (2015), available at `http://mahout.apache.org/users/recommender/recommender-documentation.html`
38. The Apache Software Foundation: Apache Solr (2015), available at `http://lucene.apache.org/solr/`
39. Thomson Reuters: Open Calais (2008-2013), available at `http://www.opencalais.com/`
40. Weiss, A.S.: Exploring news apps and location-based services on the smartphone. Journalism & Mass Communication Quarterly 90(3), 435–456 (2013)
41. Woodman, M.: Rome (2015), available at `https://rometools.jira.com/wiki/display/ROME/Home`
42. Yu, Z., Zhou, X., Zhang, D., Chin, C.Y., Wang, X., men, J.: Supporting context-aware media recommendations for smart phones. Pervasive Computing, IEEE 5(3), 68–75 (July 2006)