Combining Content-based and Collaborative Filtering for Personalized Sports News Recommendations

Philip Lenhart
Department of Informatics
Technical University of Munich
Boltzmannstr. 3, 85748 Garching, Germany
philip.lenhart@in.tum.de

Daniel Herzog
Department of Informatics
Technical University of Munich
Boltzmannstr. 3, 85748 Garching, Germany
herzogd@in.tum.de

ABSTRACT

Sports news are a special case in the field of news recommendations as readers often come with a strong emotional attachment to selected sports, teams or players. Furthermore, the interest in a topic can suddenly change if, for example, an important sports event is taking place. In this work, we present a hybrid sports news recommender system that combines content-based recommendations with collaborative filtering. We developed a recommender dashboard and integrated it into the Sport1.de website. In a user study, we evaluated our solution. Results show that a pure content-based approach delivers accurate news recommendations and the users confirm our recommender dashboard a high usability. Nevertheless, the collaborative filtering component of our hybrid approach is necessary to increase the diversity of the recommendations and to recommend older articles if they are of special importance to the user.

CCS Concepts

•**Information systems → Recommender systems;**

Keywords

Recommender System; Sports News; Content-based; Collaborative Filtering; Hybrid

1. INTRODUCTION

Recommender systems (RSs) suggest items like movies, songs or points of interest based on the user's preferences. Traditional RSs have to face some challenges when recommending such items. One of the most common problems is the cold-start problem [1]. News items without any ratings cannot be recommended while new users who did not share their preferences with the RS yet cannot receive any personalized recommendations. When recommending news, recency plays a critical role [11]. News have to be up-to-date but sometimes older articles are important if there is a connection to current events. Sports news represent only one category of news but they complicate the news recommendation process. People interested in sports are often characterized by a strong emotional attachment to selected sports, teams or players. With regard to recommendations, a user could be in favor of a lot of news about one team while she or he absolutely wants to avoid any information about a rival. Furthermore, the user's interest in a topic can suddenly change. For example, during the Fifa World Cup, even some people who are not interested at all in soccer want to be kept up-to-date with regard to current results.

In this work, we want to examine how well content-based RSs work for recommending sports news. In addition, we extend our RS by collaborative filtering. We develop a recommender dashboard and integrate it into the website of the German television channel and Internet portal Sport1[1]. We evaluate both algorithms and the usability of our prototype in a user study.

This paper is structured as follows: in Section 2 we present related work and highlight our contribution to the current state of research in content-based and hybrid news RSs. We explain how we combine a content-based and a collaborative filtering component to a hybrid sports news RS in Section 3. Our development is evaluated in a user study. The results of this study are summarized in Section 4. This work ends with a conclusion and an outlook on future work.

2. RELATED WORK

Different approaches try to tackle the problem of personalized news recommendations. One of the first news RSs was developed and evaluated by the GroupLens project [9]. The researchers used collaborative filtering to provide personalized recommendations. A seven-week trial showed that their predictions are meaningful and valuable to users. Furthermore, they found out that users value such predictions for news because in the experiment, the participants tended to read highly rated articles more than less highly rated articles. Liu et al [10] developed a news RS based on profiles learned from user activity in Google News. They modeled the user's interests by observing her or his past click history and combined it with the local news trend. Compared with an existing collaborative filtering method, their combined method improved the quality of the recommendations and attracted more frequent visits to the Google News website.

Using article keywords to build user profiles for news recommendations has already been researched. The Personalized Information Network (PIN) creates user profiles by so

[1]http://www.sport1.de/

CBRecSys 2016, September 16, 2016, Boston, MA, USA.

called interest terms which consist of one or more keywords [15]. Experiments show that PIN is able to deliver personalized news recommendations on-the-fly.

Some researchers used hybrid RS combining different techniques to suggest news articles. Claypool et al. [7] developed P-Tango, an online newspaper combining the strengths of content-based and collaborative filtering. News@hand is a system that makes use of semantic-based technologies to recommend news [5]. It creates ontology-based item descriptions and user profiles to provide personalized, context-aware, group-oriented and multi-facet recommendations. Its hybrid models allow overcoming some limitations of traditional RS techniques such as the cold-start problem and enables recommendations for grey sheeps, i.e. users whose preferences do not consistently agree or disagree with any group of people [7]. The authors evaluated the personalized and context-aware recommendation models in an experiment with 16 participants. Results showed that the combination of both models plus their semantic extension provides the best results [6]. De Pessemier et al. [8] used an hybrid approach to recommend news of different sources. Their approach combines a search engine as a content-based approach with collaborative filtering and uses implicit feedback to determine if the user is interested in a certain topic. The recommendations are presented in a web application optimized for mobile devices.

Asikin and Wörndl [2] presented approaches for recommending news article by using spatial variables such as geographic coordinates or the name and physical character of a location. Their goal was to to deliver serendipitous recommendation while improving the user satisfaction. A user study showed that their approaches deliver news recommendations that are more surprising than a baseline algorithm but still favored by the users.

To the best of our knowledge, no research focusing on the special case of sports news has been done. In this work we want to show how sports news can be recommended in a content-based approach. In addition, we extend this RS by a collaborative filtering component. In a user study, we evaluate both approaches to find out if the hybrid algorithm improves the recommendations. We show how sports news can be suggested to real users by developing and testing a fully working recommender dashboard which can be integrated into existing webpages.

3. DEVELOPMENT OF A PERSONALIZED SPORTS NEWS RECOMMENDER SYSTEM

This section explains the algorithms we used in our RS, but also illustrates the user profile modeling that is needed to provide personalized recommendations. Finally, the prototype is shown to point out how our concepts are implemented on a website.

3.1 User Profile and Preference Elicitation

The user's preferences with regard to sports news are expressed by keywords of articles that she or he is reading. Each article of our recommendation database is characterized by five to ten keywords which are automatically generated by analyzing the article's text. We are storing a list of keywords and how often each keyword occurs in articles the user has read. The more articles the user is reading, the better the recommendations are optimized with regard to the user's preferences. In our first prototype the counter for each present keyword is incremented by one when the user reads the article containing this keyword. In future works, the keywords in an article could be weighted according to the relevance and importance of the keyword to the article.

The new user problem affects every user who did not read an article yet. As explained, sports news differ from other kinds of news in the emotional attachment to selected sports, teams or players. We use this finding to overcome the new user problem. Before starting the recommendation process, the user can specify her or his favorite sport and team. News can then be recommended based on this selection and will improve when the user is reading articles, thus providing implicit feedback.

3.2 Content-based Sports News Recommendations

Content-based recommender suggest items that are similar to items the user liked in the past [1]. Since the user profile uses weighted keywords, we use vector representations of the profile and the articles to calculate the similarity between two articles.

One of the most important things for a news RS is to provide articles that are not dated. Especially in the sports news domain the environment is fast changing and usually the user is not interested in news about a sports event or her or his favorite team that are not up-to-date. The main challenge for us was to determine how old sports news can be before they are not considered for recommendation anymore. For our content-based RS we only take news into account that are not older than three days. Besides only providing relevant articles, this decision promises a better performance of the algorithm. The more articles are considered, the longer the process of calculating the recommendations takes. Our system currently uses only one news provider, but if the system grows, this could lead to a significant loss of performance. Our hybrid algorithm which incorporates collaborative filtering is also able to provide older articles if they are of high importance to the user (cf. Section 3.4).

The formula below computes the similarity between two articles (g and h),

$$sim(g,h) = \frac{\sum_{i \in W}(g_i * h_i)}{\sqrt{(\sum_{i \in W} g_i^2 * \sum_{i \in W} h_i^2)}} \quad , \qquad (1)$$

where

g,h *are* vectors representing articles with weighted keywords,

W *is* the set union of the particular keywords,

i *is* a keyword and

g_i,h_i *are* the weights of i in g and h, respectively.

In the computation of content-based similarity scores we only consider the relative dimension of the keyword weights. For the reason that user profiles have different dimensions compared to articles, the use of relative dimensions provides better results for our system. As an illustration of the main idea of the algorithm, let us consider the simple case where the user profile contains two keywords with the weights 5 and 10. Additionally there is another article with these two keywords but the weights are 1 and 2, respectively. In this case the similarity is 1, because of the same relative dimensions of the article and the user profile.

The algorithm considers every article as an element in a vector space, where the keywords are forming the base. The coordinate of an article in the direction of a keyword is given by the weight of this keyword. If the keyword does not occur, the weight will be 0.

We normalize each article relative to the standard scalar product by dividing it by its absolute value. Consequently, the standard scalar product of the two normalized vectors conforms to the desired comparison features. Even if there are negative weights, e.g. for active suppressed keywords, the algorithm calculates similarities correctly.

In order to understand the similarity calculation better, we explain how the algorithm works for an article with itself (or another article with the same weight proportions). In this case, the scalar product is 1, because of the way the vectors are normalized. But if two articles have disjunctive keyword sets, the result is 0, because such vectors are orthogonal to each other.

In the end, the system sorts the articles by similarity descending and returns the 50 articles with the highest score.

3.3 Collaborative Filtering Component

In contrast to content-based filtering, a collaborative RS uses the ratings of other users to calculate the similarity of articles [1]. Different algorithms for item-based collaborative filtering exist. We explain some common algorithms in the following and explain our choice for a sports news RS. Therefore we refer to [12] and [14].

Vector-based / Cosine-based Similarity:

$$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{\|i\| * \|j\|} \tag{2}$$

The first algorithm is the vector-based, also called cosine-based, similarity. In this algorithm, items are represented as two vectors that contain the user ratings. The similarity between item i and item j is calculated by the cosine of the angle between the two vectors. The "$\cdot$" denotes the dot product of vector $\vec{i}$ and vector $\vec{j}$ [12]. Due to the fact that cosine based similarity does not consider the average rating of an item, Pearson (correlation)-based similarity tries to solve this issue.

Pearson (Correlation)-based Similarity:

$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i}-\bar{R}_i)(R_{u,j}-\bar{R}_j)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_i)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_j)^2}} \tag{3}$$

The first part of this algorithm is to find a set of users U that contains all users who rated both items i and j. These items are called co-rated items. Not co-rated items are not taken into consideration of this algorithm. This similarity calculation is based on how much the rating of a user deviates from the average rating of this item. $R_{u,i}$ represents the rating of a user u on item i and $\bar{R}_i$ denotes the average rating of an item i.

Adjusted Cosine Similarity:

$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i}-\bar{R}_u)(R_{u,j}-\bar{R}_u)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_u)^2}} \tag{4}$$

Adjusted cosine similarity takes into account that the rating preferences of the different users differ. There are some user that always give low ratings, but on the other side there are users that rate highly in general. To avoid this drawback, this algorithm subtracts the average rating of a user $\bar{R}_u$ from each rating $R_{u,i}$ and $R_{u,j}$ on the items i and j.

The presented advantage is the reason why we apply the adjusted cosine similarity in our development. First, the system has to calculate the related articles list of all articles. To compute the related article list of an article, we iterate through the list of all articles. If the current article is not the same as the article to compare, we will calculate the similarity.

The function returns a value between minus one and one. Since the article rating range is from one to five, we map the similarity to the rating range by using the linear function:

$$sim = 2 * sim + 3 \tag{5}$$

There is one bigger problem in the adjusted cosine similarity calculation. When there is just one common user between articles, the similarity for those items is one, which is the highest value of the rating range. This is due to the subtraction of the average rating from the user's rating. To avoid the effect that the best rated articles are the articles with just one common user, we specified a minimum number of users that two articles need to have in common. In our implementation the minimum number of common users is five. When there are less than five common users, the articles are not considered in our related article list.

Afterwards, we sort the list by similarity. Moreover, we set a limit of 50 related articles to avoid additional expenses due to articles that are not considered for computation. When the related article list is calculated, we can predict the top articles for a user. For each article in the related articles list, we check if the user has already read the article. If that is the case, the article is not recommended anymore and the system jumps to the next article. If it is a new article, the prediction is calculated and added to the recommendation list. After calculating the prediction for every article, we sort the recommendation list by the predicted value.

The prediction $P_{u,i}$ can be calculated by the weighted sum method [12]:

$$P_{u,i} = \frac{\sum_{all\ similar\ items,N}(s_{i,N} * R_{u,N})}{\sum_{all\ similar\ items,N}(|s_{i,N}|)} \tag{6}$$

This approach is "computing the sum of the ratings given by the user on the items similar to i" [12]. Afterwards, each rating $R_{u,j}$ is weighted by the similarity between item i and item $j \in N$. The basic idea of this approach is to find items that are forecasted to be liked by the user. The top predicted items are recommended to the user.

A key advantage over content-based filtering techniques is the fact that collaborative RSs are able to provide a bigger variety of topics. Furthermore, with collaborative techniques, it is possible to provide event- or trend based recommendations, such as news about the World Cup. A pure content-based RS is not be able to recommend news about the darts championship if the user has just read football articles before.

3.4 Weighted Hybrid Recommender

In this section, we explain how we combine the content-based and the collaborative components to a hybrid sports news RS.

As combination technique, we use the weighted hybrid strategy as described in [4]. For our first version, we decided to weight both components equally. The content-based component is important for recommending new articles even if no ratings exist. Additionally, the content-based system is able to provide content to users with special interests as well. Moreover, the content-based version is important, because of fan culture and constant interest in some topics. But we decided that the collaborative filtering part is as important as the content-based component, due to the event-based environment and the changing popularity of some sports. We want the system to be able to recommend articles that are attractive for just a small time slot. For example, many persons are interested in the Olympic Games, but not in the different kind of sports in general.

We determined that the weights are just applied if both components of our system recommend the corresponding article. Otherwise, additional requests have to be sent to calculate a combined score for each article. If just one component recommends the article, just the score of this component is taken with the full weight. Due to this procedure, we are able to provide recommendations of both components.

3.5 Implementation

We developed a dashboard widget which can be integrated in existing websites to provide personalized sports news recommendations. For the development of the front-end, we used the JavaScript framework AngularJS, the style sheet language Less and HTML5 local storage. Figure 1 shows a current screenshot of our recommender dashboard. Nine recommendations are presented at one time. When the mouse is moved over one article, the user can read it ("Ansehen") or reject the recommendation ("Entfernen").

It is critical to identify the user every time she or he accesses the RS to provide personalized recommendations. We avoided to implement a mandatory login as this could be a big obstacle for new users visiting a sports news website. Instead, we calculate a Globally Unique Identifier (GUID) which is then stored in HTML5 local storage without an expiration date. This is an important advantage for our RS. Due to the fact that HTML5 local storage has no explicit lifecycle, we can use it not only for user identification, but also for generating a profile of the user. Storing this data on client site is decreasing the amount of data stored on the server which makes the system more scalable. Only the item similarities and recommendations are calculated on the server, due to direct access to articles from our backend.

In order to get content-based recommendations, the client sends an Ajax request to a NodeJS server. Therefore, the user ID and the corresponding profile are sent as parameters. We decided to use an Ajax request due to the fact that the computation causes no overhead at site loading if it is done from JavaScript code. At our backend, the weighted keyword profile is sorted by the keyword name alphabetically. As mentioned, we receive articles published within the last three days. After obtaining those articles from our article repository, we add the suitable keywords to each of them. The weighted keywords are in the same form as the user profile to make them comparable to each other. The system calculates the similarity of the user profile with every article. Therefore, the union of the keyword sets is built. Subsequently, the similarity is computed using formula 1.

A JSON response sends the 50 articles with the highest score back to the client. The response is then processed by the Angular directive of the personalized dashboard. If the user removes an article, the next recommended article takes over its place. In addition, further statistics like the last read articles or last and next matches of the preferred team are displayed.

For the computation of collaborative recommended articles we use the same NodeJS server. In contrast to other systems, we do not store our data in a database. Due to the fact that we have to iterate through lists most of the time to compare ratings and users, we decided to use arrays to store our data within the application. The ratings provided by the user are collected in a rating variable that is kept in memory. It stores JSON objects with the user ID, the article path and the provided rating. Furthermore, the current date is used to distinguish current data from dated ratings that are not relevant for our system anymore.

To speed up the similarity computation, we adapt the average rating of a user every time providing a new rating. The average ratings are kept in an extra variable for performance reasons. The current average rating and the number of ratings provided by the given user is enough to adapt the average. Just a few basic arithmetic operations are necessary to avoid calculating the average from the rating variable every time from scratch. We minimize the accesses to the rating variable due to the fact that this variable is the main component of our server. Most of the requests read or write this variable. Every variable access that can be eliminated helps to improve the system's performance.

Moreover, we store a list of users as well as a list of articles to iterate through these arrays without generating them first. Using a list of all articles is primarily important when the system computes related articles. The list of related articles is updated every hour. A cronjob is executed every hour to consider current news as well. After one hour there are more ratings provided and the new item problem of a pure collaborative RS is suppressed.

For similarity calculation of two items, we need to find a set of users that contains all users who rated both items. Therefore, we generate a list of objects that contain the articles and all the users who rated the corresponding article. To compute the user set of two articles, we compare the two user lists and determine the intersection.

The combination of the content-based and the collaborative part of our RS is implemented in JavaScript. First, we send an Ajax request to our backend to collect the content-based recommended articles. In addition, another request is sent to our NodeJS server where the collaborative filtered articles are computed. If the collaborative filtered recommendations are returned correctly, the system computes the combination of both article sets. Finally, the recommended articles are returned and the JSON response is sent to the application.

In the news domain the age of an article is definitely one of the most important properties when the article's attrac-

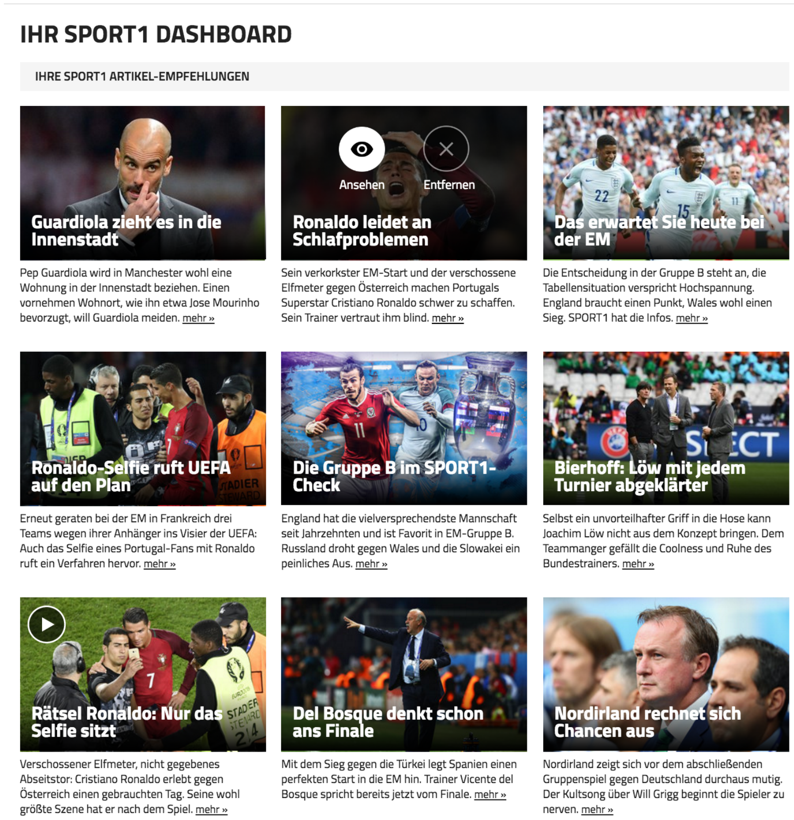

Figure 1: Recommender Dashboard

tiveness is determined for a user. Because of the recency problem, we decided to implement a route in our NodeJS server to remove dated ratings and articles from our system. Every two weeks a cronjob is executed and every rating that is older than four weeks is removed from the ratings table. The removal of those ratings implies the secondary effect that old users that do not exist anymore are removed as well. This is a very common scenario in our system, due to the fact that we identify the user by using HTML5 local storage. If the local storage is deleted, the old user ID does not occur anymore. We decided to use these time intervals, because our content-based version considers only articles published the last three days and we want to provide recommendations of articles older than a few days as well if an older article is getting popular again. In this case, our system is able to recommend those articles as well as long as ratings are provided in the last four weeks.

4. EVALUATION

We conducted user studies to evaluate our algorithms and the usability of our recommender dashboard. In this section we present the goals, the procedure and the results of our evaluation. We interpret our findings to answer the question how well content-based algorithms support user in receiving interesting sports news and if a hybrid algorithm can improve the performance of our RS.

4.1 Analysis of Usage Data of the Content-based Recommender

In order to collect usage data of real users, we tested the content-based approach on the live version of the Sport1 website. For this purpose, the recommender dashboard prototype is presented to one percent of the users. Due to the fact that the website is visited by thousands of users every day, one percent of the users is enough to evaluate not only the functionality but also the usability of our RS. In future, we will increase the amount of test users from time to time and adapt our implementation accordingly. We used Google Analytics to measure relevant Key Performance Indicators (KPIs) that help us to evaluate our solution. We analyzed how much the users clicked on the read and the remove button, respectively. Moreover, we tested how often the users navigated to articles they have already read by using the last read articles widget. In addition to the event tracking, we analyzed if there is an impact on the article ratings due to the new personalized dashboard. This is why we compare the average ratings of different articles. Articles are just taken into account, if they are rated by the one percent of users that can use the personalized dashboard.

At the end of our live study 5132 user IDs were registered on our server. This does not mean that more than 5000 different users used the dashboard due to the fact that every device has its own GUID and if the history of the browser is deleted, a new ID is generated. But there were enough users producing events we can track.

The click behaviors of the users give information about

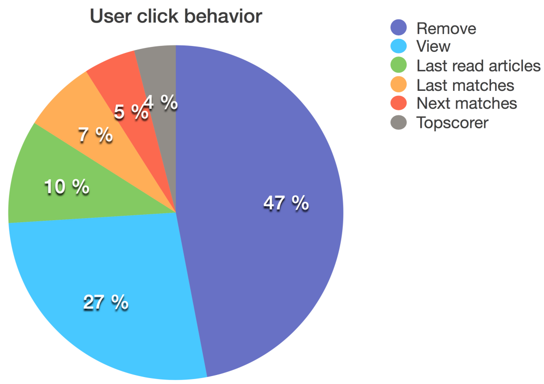

Figure 2: Analysis of the user's click behavior

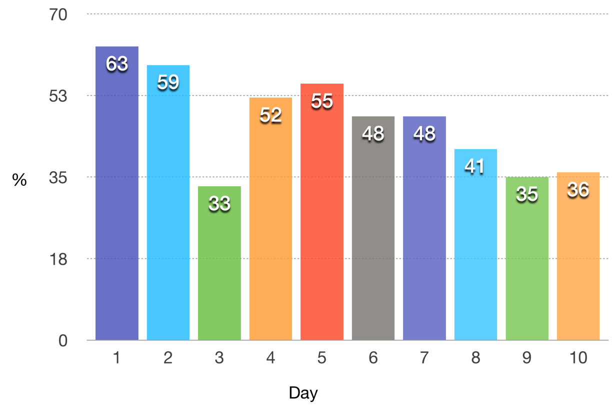

Figure 3: Daily clicks on remove (figures in per cent)

the user acceptance of the different components. Figure 2 illustrates how many clicks are executed on the different components of the personalized dashboard.

Almost 50 percent of the clicks were executed on the remove button of the news recommendation widget. On the first view, this number is quite high. But if we consider that at all the other buttons navigate the user to another page, it is obvious that the remove button is executed more often than all the other buttons. If the user clicks on remove, the article will disappear and a new one will be displayed in the dashboard. The user is then able to interact again with the dashboard. 27 percent of the clicks are executed on the view article button, which is a good proportion. Especially, if we consider that the RS is new, it is noticeable that after every third interaction, an article that potentially fits to the interests of the user, is recommended. To get better informations about the quality of the recommendations, we need to organize a long-term study. The sports news domain is very dynamic and the click behavior is changing depending on the current events. By that reason, the two week evaluation is not enough to ensure that the amount of clicks on an element is constantly similar. Around one quarter of all clicks are executed on links and buttons which are not part of the recommender dashboard but provide additional information such as last read articles and team-related statistics such as last and next matches and top scorers.

We expect that the quality of the recommendations increases with the time of use. To test this assumption, we analyzed the trend of the remove button clicks. Except for some days, the ratio of clicks on remove decreased with every day performing our testing (cf. Figure 3). The exceptions may base on new users or users that do not read many articles on the website. If none or just a few articles are read before using our dashboard, the quality of the recommendations will be low. Since the dashboard is just presented to one percent of the users, we are not able to give evidence that the subjective quality will be the same when publishing the dashboard to all users. With increasing the number of testers, the ratio of remove button clicks increases at the beginning and then falls again with the time of use. We detected this when we released the dashboard on the website.

To analyze if the dashboard has an effect on the article ratings, we compared three types of articles. First, articles that are bad rated in general, second average rated articles and finally articles that are high rated in general. A generally bad rated article gets a better score from our RS users. This is mainly due to the fact that we use the implicit feedback of three stars if a user reads the article. Moreover, the bulk of the users is not providing any rating for an article. So the average rating of the testers is almost at three stars for bad and average rated articles. For high rated articles, the RS users scored a little lower in general. The chance to get such an article provided by our system is higher due to the fact that more comparable users are available for the most read articles. If the user clicks on remove, the lowest rating of one star is implicitly provided and the average rating is decreasing. Since the personalized dashboard is not established on the website, we can sum up that the recommendations have almost no effect on the rating scores. This may change if the users will use the dashboard as their first contact point on the website.

It was also noticeable that the users want to read already read articles again. The last read article widget helps them to navigate back and easily get an overview of the last interactions. We expect that the amount of clicks in this widget will decrease in the future. Since new articles are potentially more attractive for a user, we can not imagine that every tenth click is executed on an already read article. We believe that the users in the study were curious and wanted to test this new feature.

4.2 Comparison of Content-based and Hybrid Recommendations

Our hybrid algorithm extends the presented content-based approach by a collaborative filtering component. This algorithm is not part of the live version of the RS yet. We tested the hybrid recommendations with a selected user group. We paid attention to choose persons from different backgrounds and with diverging interests to ensure comparability to our users.

The participants had to rate the RS in two scales on a scale from 1 (worst rating) to 5 (best rating): How well the recommendations fit their interests and how diversified they are. The pure content-based solutions served as a baseline algorithm. In total, we received 40 completed questionnaires for the content-based approach and 20 for the extended, hybrid RS.

The results show that the recommendations are not diversified enough in our pure content-based approach (Ø2.9), but they improve in our hybrid implementation where the average rating was 3.3. The content-based recommendations are representing the interests of the user (Ø3.4) which shows us that the dashboard provides additional value. This value did not change in our hybrid version. It was noticeable that the more frequent a user is visiting the website, the more he is satisfied with the result of the recommendations. The users that visited the website every day gave an average rating of 3.6. This confirms that the quality of our recommendations increases over time.

4.3 Usability of the Recommender Dashboard

Besides evaluating our recommendation algorithms, we asked the study participants to rate the usability of our recommender dashboard. We used the well-established System Usability Scale (SUS) [3]. This questionnaire consists of ten questions providing a global view of subjective assessments of usability. Participants respond using a Likert scale with five response options; from Strongly agree to Strongly disagree. Furthermore, our participants were allowed to add further thoughts in a free-text field.

To calculate the SUS score, the answers for each question are converted to a new score from 0 to 4 where 4 is the best score and 0 is representing the worst possible answer of this question. Afterwards the different scores are added together and multiplied by 2.5 to get a ranking value between 0 and 100 [3]. Every SUS score above 68 is considered as above average, everything lower than 68 as below average [13]. The average scores of each question, collected in our user feedback, are shown in Table 1.

The score is calculated by adding the scores and multiplying the sum with 2.5:

$$score = sum * 2.5 = 34.8 * 2.5 = 87 \tag{7}$$

The score of 87 exceeded our expectations although we attached great importance to the design and usability of our system. This was required because the dashboard is implemented on the live website of Sport1. Nevertheless the users mentioned some desires concerning the usability. For example, some users wished to change the design by choosing their own colors. Since these informations have not an direct impact on our RS implementation, we will not deepen these suggestions here.

4.4 Discussion

The user study results show that our content-based RS is a promising approach to suggest sports news to users. The recommendations fit the users' interests and improve when the users provide more feedback. Nevertheless, the diversity of the recommended articles remains to be low. This is a typical problem of pure content-based RS and can be overcome by using a hybrid solution. We extended the RS by a collaborative filtering component which increased the diversity of the recommendations.

As described before, it is very important that a news RS provides current articles. Especially in sports, the environment is very dynamic and the news topics are changing all the time. For that reason the system can not be a pure collaborative RS. With collaborative filtering it is almost impossible to recommend new items. But this problem can be solved by using a content-based component as well. Content-based RSs can provide content that fits to the general interests of the users. In addition, attention should be paid to event based interests, e.g. the Super Bowl, an event that is closely followed by many people. If a user has no interests in American Football in general, the content-based RS does not provide articles about the Super Bowl. So there must be a combination of both techniques to benefit from the strengths of each component.

5. CONCLUSION AND FUTURE WORK

In this work, we tackled the problem of recommending sports news. Sports news are a special case in the field of news recommendations as users often come with a strong emotional attachment to selected sports, teams or players. Furthermore, the interest in a topic is event-driven and can suddenly change. We developed a content-based RS that creates user profiles based on implicit feedback the user shares when reading articles. Using automatically created keywords, the similarity between articles can be measured and the relevance for the user can be predicted. This approach delivers accurate recommendations but lacks diversity. In a first prototype, we designed and evaluated a hybrid algorithm that extends our content-based RS by a collaborative component. This hybrid approach increases diversity and also allows to recommend older articles if they are of particular interest for the user.

To improve the quality of the hybrid recommendations, we will adjust our implementation from time to time and test if the adaptions serve their purpose. First, we will test different weights for the two components. One idea is to increase the weight of the content-based version. The decrease of the weight of the collaborative version does not exclude the event-based recommendations. Even if the collaborative part does just count one third, it is able to provide recommendations because if the article is only recommended by our collaborative version, just the score of this component is taken into account. If both components provide this article recommendation, the content-based version is more adapted to the users interests. To find out which weight ratio is the best for our case, we have to analyze the implicit and explicit user feedback for a longer time period. The evaluation of the weights is just meaningful if the feedback is collected for a few months to avoid temporally fluctuation, which is quite common in the news domain.

Furthermore, we want to implement a switching hybrid as well. If there is a new item, the collaborative filtering method can not provide recommendations from the first second. This is the strength of our content-based version. The RS has to switch to the content-based version if the article is newer than a specific date. Recommendations for a new user are calculated by our collaborative filtering component to handle new users as well as the preferences at the first use of the system are not sufficient to compute pure content-based recommendations. If a larger user profile is constructed and an article is not published in the last minutes, the combination of both techniques will be applied as described before.

We tested our first developments in a two-week user study. Our content-based RS has been tested with live users while the hybrid approach was only accessible for a selected user group. In future, we want to conduct larger studies with more users for all algorithms we develop. Our first results will serve as the baseline for future extensions and other algorithms.

Table 1: Questions and Results of the SUS questionnaire

Number	Question	Average Score
1	I think that I would like to use this system frequently.	3.4
2	I found the system unnecessarily complex.	3.3
3	I thought the system was easy to use.	3.6
4	I think that I would need the support of a technical person to be able to use this system.	3.9
5	I found the various functions in this system were well integrated.	3.3
6	I thought there was too much inconsistency in this system.	3.1
7	I would imagine that most people would learn to use this system very quickly.	3.6
8	I found the system very cumbersome to use.	3.6
9	I felt very confident using the system.	3.2
10	I needed to learn a lot of things before I could get going with this system.	3.8

6. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[2] Y. A. Asikin and W. Wörndl. Stories around you: Location-based serendipitous recommendation of news articles. In *Proceedings of 2nd International Workshop on News Recommendation and Analytics*, 2014.

[3] J. Brooke. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, pages 189–194, 1996.

[4] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Nov. 2002.

[5] I. Cantador, A. Bellogín, and P. Castells. News@hand: A semantic web approach to recommending news. In W. Nejdl, J. Kay, P. Pu, and E. Herder, editors, *Adaptive Hypermedia and Adaptive Web-Based Systems: 5th International Conference, AH 2008, Hannover, Germany, July 29 - August 1, 2008. Proceedings*, pages 279–283. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[6] I. Cantador, A. Bellogín, and P. Castells. Ontology-based personalised and context-aware recommendations of news items. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '08, pages 562–565, Washington, DC, USA, 2008. IEEE Computer Society.

[7] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR workshop on recommender systems*, volume 60. Citeseer, 1999.

[8] T. De Pessemier, S. Leroux, K. Vanhecke, and L. Martens. Combining collaborative filtering and search engine into hybrid news recommendations. In *Proceedings of the 3rd International Workshop on News Recommendation and Analytics (INRA 2015) co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 20, 2015.*, pages 14–19, 2015.

[9] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: Applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, Mar. 1997.

[10] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, 2010.

[11] Ö. Özgöbek, J. A. Gulla, and R. C. Erdur. A survey on challenges and methods in news recommendation. In *Proceedings of the 10th International Conference on Web Information Systems and Technologies*, pages 278–285, 2014.

[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM.

[13] J. Sauro. Measuring Usability with the System Usability Scale (SUS), 2011. Retrieved June 20, 2016 from http://www.measuringu.com/sus.php.

[14] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Hindawi Publishing Corporation*, 2009, 2009.

[15] A.-H. Tan and C. Teo. Learning user profiles for personalized information dissemination. In *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on*, volume 1, pages 183–188, May 1998.