Offline and Online Evaluation of News Recommender Systems at swissinfo.ch

Florent Garcin
Artificial Intelligence Lab
Ecole Polytechnique Fédérale de Lausanne
Switzerland
firstname.lastname@epfl.ch

Boi Faltings
Artificial Intelligence Lab
Ecole Polytechnique Fédérale de Lausanne
Switzerland
firstname.lastname@epfl.ch

Olivier Donatsch
SWI swissinfo.ch
Swiss Broadcasting Corp.
Switzerland

Ayar Alazzawi
SWI swissinfo.ch
Swiss Broadcasting Corp.
Switzerland

Christophe Bruttin
SWI swissinfo.ch
Swiss Broadcasting Corp.
Switzerland

Amr Huber
SWI swissinfo.ch
Swiss Broadcasting Corp.
Switzerland

ABSTRACT

We report on the live evaluation of various news recommender systems conducted on the website *swissinfo.ch*. We demonstrate that there is a major difference between offline and online accuracy evaluations. In an offline setting, recommending most popular stories is the best strategy, while in a live environment this strategy is the poorest. For online setting, context-tree recommender systems which profile the users in real-time improve the click-through rate by up to 35%. The visit length also increases by a factor of 2.5. Our experience holds important lessons for the evaluation of recommender systems with offline data as well as for the use of the click-through rate as a performance indicator.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

Keywords

recommender system; news; real-time; live; evaluation

1. INTRODUCTION

The proliferation of online news creates a need for filtering to focus on interesting articles. Compared to other products, however, recommending news has specific challenges [7, 22]: news preferences are subject to trends, users do not want to see multiple articles with similar content, and frequently we have insufficient information to profile the reader.

We addressed these challenges in our previous research by introducing a new class of news recommender systems based on Context Trees (CT) [7, 8]. This class of recommender systems adapts its model to current trends and reader preferences. The model evolves with the read articles such that it is always up to date and recommendations are generated in real-time.

Previously, we evaluated the context-tree recommender systems in an offline setting and showed that there exists a trade-off between accuracy and novelty of recommendations [7]. However, this trade-off can only be answered by performing a live evaluation. This led us to several open questions: Which recommender systems should be deployed in practice? Do CT recommender systems improve the click-through rate over baseline methods? Do recommender systems increase the time spent by the users on the news website? Is there a major difference between offline and online evaluation? Although we were able to find elements of answers in previous works [12, 15, 18, 20], they do not address the context-tree technique and so we performed our own live investigation on the news website *swissinfo.ch* operated by the Swiss Broadcasting Corporation.

In this paper, we report on the live evaluation of CT recommender systems. We show that CT systems significantly improve the click-through rate (CTR) on swissinfo.ch, and also increase the visit length. We also learn important lessons regarding the usefulness of offline evaluation: the relative performance predicted on offline data was in fact exactly the reverse order of what was actually observed on the live system. Furthermore, we show that the click-through rate is not always a good indicator of actual impact.

2. RELATED WORK

Although the first recommender systems were originally designed for news forums [17], the literature describing actual implementations and evaluations on live news websites is not common compared to the general literature on recommender systems [6, 12, 15, 18, 20].

Liu et al. [15] analyse the deployment of a hybrid recommender system on the news aggregator Google News. They compare their method against the existing collaborative filtering system implemented by Das et al. [6], and consider

RecSys'14, October 6–10, 2014, Foster City, Silicon Valley, CA, USA.
Copyright 2014 ACM 978-1-4503-2668-1/14/10 ...$15.00.
http://dx.doi.org/10.1145/2645710.2645745.

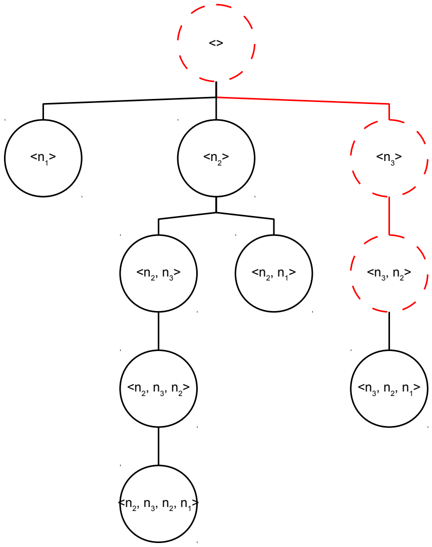


Figure 1: Context tree built with the sequence $\langle n_1, n_2, n_3, n_2\rangle$. Nodes in red-dashed are contexts matching the new sequence $\langle n_2, n_3\rangle$.

only logged-in users for the evaluation. They show a 30% improvement over the existing collaborative filtering system.

Kirchenbaum et al. [12] conduct an online evaluation with logged-in users of several recommender systems for news articles on Forbes.com. They report that a hybrid system performs the best, with a 37% improvement over popularity-based methods.

Said et al. [18, 20] study the weekly and hourly impressions and click-through rates in the Plista news recommender framework [11], which delivers recommendations to multiple news websites. They observe that live evaluation is sensitive to external factors not necessarily related to recommendations. They also identify trends in recommendations related to the type of news websites (traditional or topic-focused news sources). Readers of topic-focused websites are less likely to take recommendations than readers of traditional news websites [18]. Unfortunately, it is not clear which recommender algorithm is used in the Plista framework and their analysis.

Our evaluation is similar to the one of Kirchenbaum et al. [12], but differs in two crucial points. First, we consider anonymous users who are impossible to track across multiple visits. Second, the nature of the websites (Forbes.com and swissinfo.ch) is not the same, and thus readers' behaviours are different [18].

3. RECOMMENDER SYSTEMS

In our previous research [7], we have developed a new class of recommender systems based on Context Trees (CT). CT recommender systems create context-dependent recommendations by assigning a local prediction model to each context. We briefly recall how CT recommender systems work.

A context-tree recommender system builds a hierarchy of contexts, arranged in a tree such that a child node completely contains the context of its parents. A context can be the sequence of stories read by a user, the sequence of topics, or of topic distributions.

The root node corresponds to the most general context, i.e. when no information is available to profile the user. In that case, the recommendations are generated based on the most popular or most recent stories for instance. However, the more the user browses the stories, the more contexts we able to extract, thus building a deeper context tree. This results in generating finer-grained recommendations, not only based on the most popular or most recent stories.

For instance, the sequence $\mathbf{s}_u = \langle n_1, n_2, n_3, n_2\rangle$ is the ordered list of news stories $n_i \in \mathcal{N}$ read by a user u (her history). The corresponding context tree is depicted in Figure 1. A new user v with the browsing sequence $\mathbf{s}_v = \langle n_2, n_3\rangle$ matches 3 contexts in that tree: the root context $\langle\rangle$ corresponding to an empty history, the context matching $\langle n_3\rangle$ which corresponds to the latest read story by user v and the context matching $\langle n_3, n_2\rangle$ corresponding to the complete history of user v.

Each context has a local prediction model. For instance, a particular model gives recommendations only for users who have read a particular sequence of stories, or users who have read an article that was sufficiently close to a particular topic distribution. We consider three possible models. The first model, called *std*, ignores the temporal dynamics. The second model, called *pop*, assumes that users are mainly looking at popular items, and the last model, called *fresh*, that they are interested in recent stories.

Recommendations are made by mixing the predictions of each local model for the given matching contexts. If a context is making poor recommendations, the CT recommender system decreases its influence by adjusting automatically the weight of the considered context.

A CT recommender system fits better the dynamic domain of news recommendations because it adapts its model to current trends and reader preferences. The model for recommendations changes along with the read articles, using a dynamically evolving context tree. In that way, the model is always up to date and recommendations are generated in real-time. In addition, CT recommender systems require only one tree and thus scales very well. They are not restricted to the history of logged-in users, but considers a one-time session for recommendation, where users do not log in.

Thanks to offline evaluations, we have shown that there is a trade-off between accuracy and novelty of recommendations, and CT recommender systems address it. In this work, we would like to explore how CT recommender systems perform on a live traffic website and how they compare to standard baselines. Thus, in addition to CT recommender systems, we consider the following baselines:

Most Popular recommends a set of stories with the highest number of clicks among the last read news items. This strategy is commonly implemented on most news websites. It does not recommend any new items, but only the ones that a reader would have already seen on the front page.

Random recommends a set of stories at random. This strategy has the advantage of recommending very diverse and potentially novel items.

There exist many news recommender systems [1, 2, 3, 10, 14], however we decided to select only these baselines for the following reasons. First, the most popular and random strategies are de-facto standards, very easy to implement and compare performance. Researchers can easily compare their results to ours relative to these baselines. Second, reproducing previous research takes time and correctly implementing an existing approach is tricky, even if it is well described. Third, most of these algorithms have not been tested on live traffic websites and thus there is no guarantee of scalability and real-time requirements. Last, we need to limit ourselves to compare simultaneously 3 to 4 algorithms. More algorithms would dilute the clicks and would not guarantee statistical significance of the evaluation.

4. EVALUATIONS

Since July 2013, we are evaluating CT-based recommender systems with live traffic on the news website *swissinfo.ch*. swissinfo.ch is a 10-language news website owned by the Swiss Broadcasting Corporation. Its content is produced specifically for an international audience with interests in Switzerland. swissinfo.ch gives priority to in-depth information on politics, society, business, culture and science & technology. It has about 3.9 million clicks per month. We have deployed recommendations only on the English version of the website.

All recommender systems are deployed with the PEN recsys framework [8], specifically designed to conduct multivariate online evaluation of news recommender systems.

With the online evaluation, we want to address the following questions:

Question 1. *Do context-tree recommender systems bring an improvement in click-through rate over baseline methods?*

Question 2. *Do recommendations in general, and recommendations made by context-tree systems in particular, increase the page views by the user on the news website?*

Question 3. *Among all CT-based recommender systems, which version and set of parameters are optimal?*

The answers to these questions are not trivial, and are specific to each website. Similar to our study, the results and conclusion of the evaluation on Forbes.com [12] are unique to this website. In addition, the reasons behind each study are not the same.

Before conducting a live evaluation, we evaluated the behaviour of recommender systems on an offline dataset in order to forecast which methods would work the best and understand how the parameters influence the performance.

4.1 Offline

Only a small set of recommender systems can be evaluated on a live website, and it has to be carefully selected. An offline evaluation helps us to select the recommender systems and their parameters that are the most likely to bring good recommendations on swissinfo.ch. To do so, we had access to a 3-weeks "recommendations-free" clicks log from swissinfo.ch. During these 3 weeks, the swissinfo.ch website did not have any recommender systems deployed such as popularity-based methods ("top 10 popular items"). The log contains 227'831 clicks on 28'525 stories.

There has been a lot of discussion on the best way of evaluating recommender systems [9, 21]. For our offline evaluation, we use visit histories from the swissinfo.ch website, and we can evaluate how well our recommendations match the news items that readers selected themselves. It is clear that this is a somewhat inaccurate measure: *a)* the user may not have liked all the items she visited; *b)* the user may have preferred one of the recommended items to the one she clicked, so the fact that a recommended item was not visited does not mean the recommendation is bad. However, it is common in recommender systems research to use the prediction of the visit history to *compare* the performance of different techniques, and select parameters for the live evaluation.

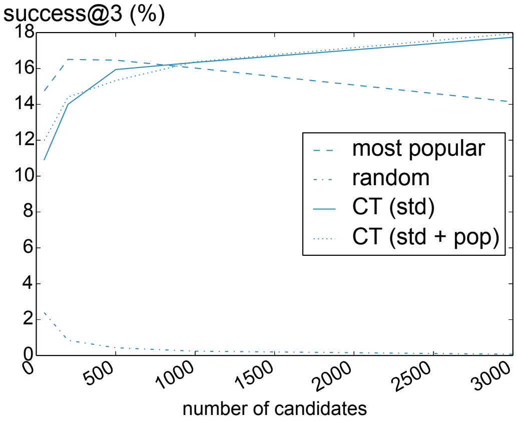
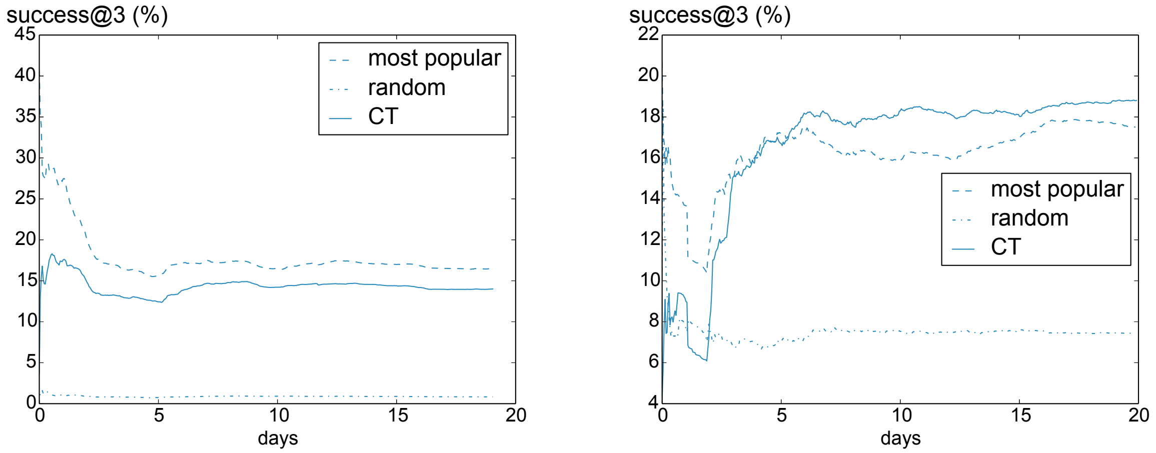


Figure 2: Offline *predicted* accuracy for different sizes of candidate set.

For a given recommender system and some fixed set of parameters, we imitated the online environment by sending clicks in the same sequence as saved in the log to the PEN recsys framework, and recorded the performance. Based on the articles last read by the user, the recommender system generates 3 recommendations. The recommendation is successful if one of these three recommendations is indeed the one that the user reads next, a measure we call *success@3*. While there are many different metrics to assess the performance of a recommender systems [5, 9], we focus on this metric because it can also be measured online and thus allows a comparison.

In addition to the baselines that recommend most popular and random items, we implemented a context-tree system with the *std* model, and one with a mixture of *std* and *pop* models. We were not able to add the *fresh* model because the log does not contain information about freshness.

Figure 2 illustrates the predicted accuracy for various sizes of the candidate set. When the number of candidates increases, the performance of the random strategy drops because interesting stories are diluted in the set. There is an optimal size for recommending the most popular items around 200 candidates. Increasing the number of candidates after this point does not improve the performance for the most popular strategy. However, CT recommender systems take advantage of more candidates to bring better recommendations. In the rest of the evaluations, we decided to pursue a conservative approach where we select parameters such that baselines are optimal, and we pick 200 as size of candidate set. This conservative choice of parameters is not in favour of CT recommenders but in that way, the performance of CT recommenders can only be improved. Note



(a) Offline *predicted* accuracy

(b) Online *actual* accuracy

Figure 3: Accuracy of online (bootstrap + phase 1) and offline evaluations.

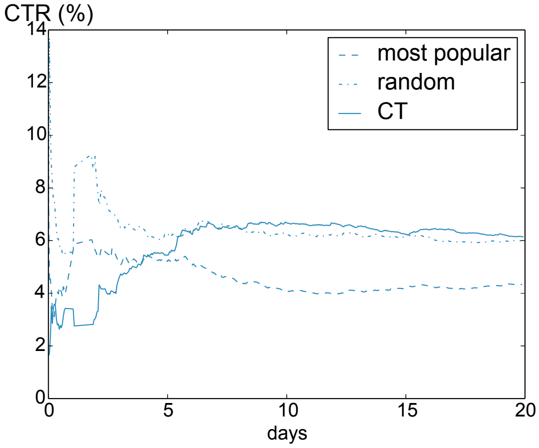


Figure 4: Online *actual* CTR (bootstrap + phase 1).

that although the websites are not the same, the behaviour of these recommender systems is similar than the one reported in our previous research [7].

Figure 3(a) shows the predicted accuracy over time. In this figure, we only implemented the *std* model in the CT recommender as it performs equally good as when we add the popular model. It takes about one week to stabilize the performance. As expected [7], recommending the most popular items outperforms other strategies. The random strategy is the poorest with an accuracy close to 1%.

4.2 Online

The online evaluation is split in two phases where we test different strategies against each other. The first phase consists of comparing a standard context-tree system against the baselines in terms of click-through rate (Question 1) and page views (Question 2). The second phase investigates different versions of CT recommender systems (Question 3). The recommenders together received 172'013 clicks on 10'653 stories in the first phase and 285'572 clicks on 10'830 stories in the second phase.

For each evaluation phase, we bootstrapped the recommender systems during one week, and evaluated them on the next two weeks. We estimated this time frame based on the previous offline evaluation, the expected traffic volume, and the time required to learn the model. It is crucial to run as many methods as possible in parallel to avoid biases in the evaluation due to trends and variations in the candidate set. However, we had to split the evaluation because we could not run more than 4 recommender systems in parallel in order to get statistical significant results.

The PEN recsys framework assigned randomly one recommender system to each visit, performing a multivariate evaluation. Note that visits are anonymous and we were not able to track a user across multiple visits. For all recommender systems, the candidate set was composed of the last 200 clicked items and 30 fresh items. We chose these conservative parameters based on our previous offline analysis. When a recommender system receives one click, it returns 3 recommendations.

4.2.1 Phase 1

For the first phase, we selected the context-tree recommender system with the standard model. For the baselines, we implemented the strategies that recommend the most popular stories and that recommend stories at random.

We evaluate the online performance by two different measures. Figure 3(b) shows the percentage of times that the 3 recommendations contain the article the user reads next, whether through a click on the recommendation or somewhere else. It is thus analogous to the success@3 metric used in the offline setting. Figure 4 shows the click-through rate, i.e. the percentage of times that the user clicked on a recommended article. The click-through rate is a widely used metric for evaluating success of recommendation, in particular in online advertising.

First consider the click-through rate (Figure 4). We can observe a reversal of the relative performance in comparison with the offline prediction. The strategy of recommending most popular articles performs the poorest by barely reaching a CTR of 4%. However, the CTR of the random strategy is now comparable to the one of the CT recommender, and it takes about 7 days for the CT recommender to reach the same CTR. Note that Figure 4 also illustrates nicely the time required (bootstrap phase) by the context-tree system to learn the behaviour of the users.

Now consider the success@3 measure (Figure 3(a) and Figure 3(b)). In contrast to the click-through rate, it measures the actual change that the recommender system brings to the user's behaviour, since it measures how recommendation influences the articles actually read. We observe that for the strategy of recommending the most popular items, the success rate increases by about 1% from 16.5 to 17.5%. Thus, about three quarters of the 4% clicks generated by the recommendation come at the expense of clicks that the user would have made elsewhere, and there is little use of having the recommender at all. This supports criticism by Zheng at al. [23] that the click-through rate for popular items is inflated by their popularity.

In contrast, for the CT recommender, readership of the recommended articles is increased by 5% from 14 to almost 19%. Here, about 5/6 of the 6% clicks on the recommendations actually increase readership of the recommended articles, and the recommender has more significant use. Even more remarkable is that recommending at random has an even bigger increase of the success rate from about 1% to over 7%, so here all of the 6% clicks actually go to increasing readership on the site.

The fact that random recommendations perform better than the recommender system is disturbing but actually explained by the fact that most users have a very short visit history (see Figure 5) and thus do not allow the recommender to make intelligent recommendations. Figure 6 shows the click-through rate per recommender system and over the length of the visit. Context-tree recommenders slightly outperform the baseline methods in general. When the visits are very short (exactly 2 articles), recommending random stories hardly increases the click-through rate (7.7%). However, for medium and long visits, the context-tree system greatly improves the CTR. For medium-length visits, the CT system has a 10.8% CTR while the random baseline is at 8.9%. For long visits, CT system is at 13.1% while the random baseline is at 9.7%. The strategy of recommending the most popular articles is definitely not the best solution: regardless of the visit length, the click-through rate stays below 7%. This is probably because users will have already read these stories elsewhere. Note that the click-through rate increases with visit length since the visitors with a longer history are also more interested users that click more in general.

When the visit length increases, CT recommenders outperform baseline methods because they take advantage of the personalized sequence of each user to create an accurate profile. No matter what the user's interest is, the baseline strategies will always perform the same. Overall, context-tree recommender system brings an 11% improvement over recommending random items, and more than a 20% improvement when considering medium to long visits (21% and 35% for medium and long visits respectively).

Another way of measuring the impact of a recommender system is to consider the influence on the average visit length as measured by the number of articles read by a user (Question 2). We break the visits into two groups: visits containing at least one click on a recommendation, and visits without any click on recommendations. Table 1 summarizes our findings. The length of the visits without taking a recommendation averages 1.25 clicks, while visits with at least one click on a recommendations tripled, with an overall average at 3.75 clicks.

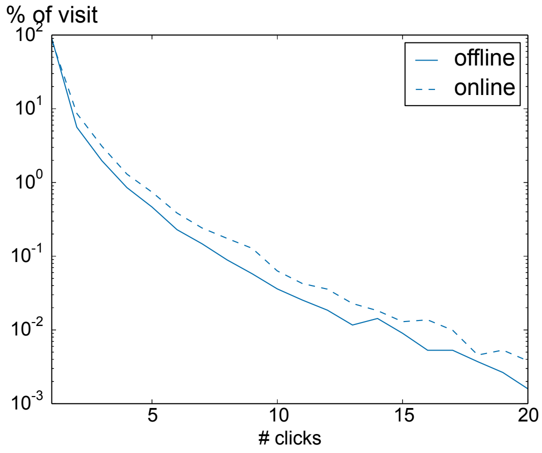


Figure 5: Distribution of visit length (in percent of the total number of visits) for online and offline evaluations.

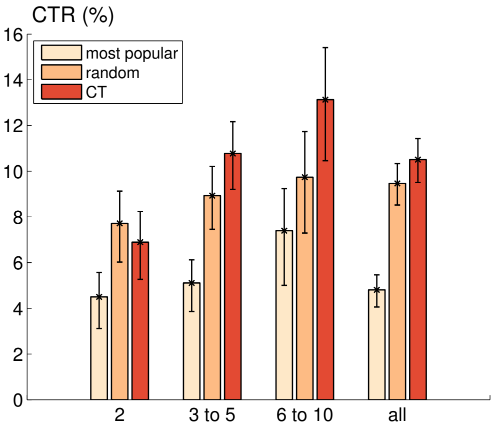


Figure 6: Online *actual* CTR over visit length, with confidence intervals at 95%.

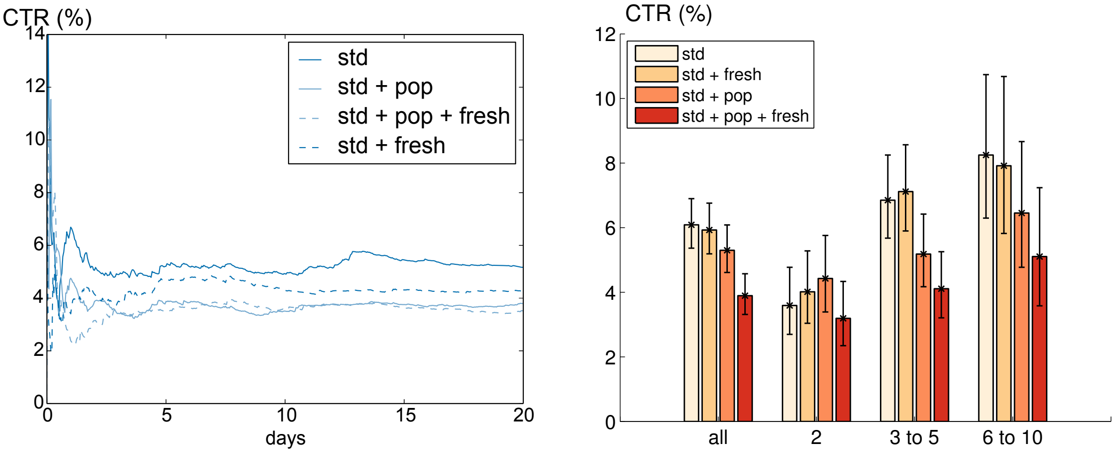
Most sessions are very short (Figure 5). Users reach the swissinfo.ch website through a search engine or a news aggregator, read one article, and leave. If we remove such behaviour by considering visits of length at least 3, the visit length increases by about 1 click, from 4.31 to 5.22 clicks. 90.4% of the visits have only one click, while with recommendations it drops to 85.0% of the visits. It is possible that users who click on recommendations are more likely to have longer visits. Since we are not tracking the users, but only their current session, it is difficult to answer this question. In Forbes.com for instance, this tendency has been observed [12].

4.2.2 *Phase 2*

In the second phase, where we compare different versions of CT recommenders (Question 3), the overall click-through rate of recommendations was slightly lower than during the

Table 1: average visit length with and without recommendations

Recommender system	w/o recommendations	w/ recommendations
Most Popular	1.25	3.36
Random	1.25	3.93
CT	1.25	**3.96**



(a) bootstrap + phase 2

(b) Phase 2 only, with confidence intervals at 95%.

Figure 7: Online *actual* CTR of different mixtures of experts for phase 2.

first phase at 5.0%. Figure 7 illustrates the click-through rate with different mixtures of models. Again, we need a bootstrap phase of about one week until the model is correctly learnt. Incorporating the popularity model decreases the click-through rate, except when the visit is very short. This is not surprising since users with a very short visit are mostly interested in what is displayed on the main page of the website, thus reinforcing popular stories. However, when the length increases, users are no longer interested in popular stories. It is not clear whether the fresh model improves recommendations. However, this might be due to the fact that the number of fresh items considered here (30) is not optimal. We believe that tuning this parameter will improve the accuracy of recommenders with the fresh model.

5. DISCUSSION

In this section, we discuss the discrepancies between offline and online evaluation, and important factors that influence directly the recommendations.

5.1 Online vs Offline

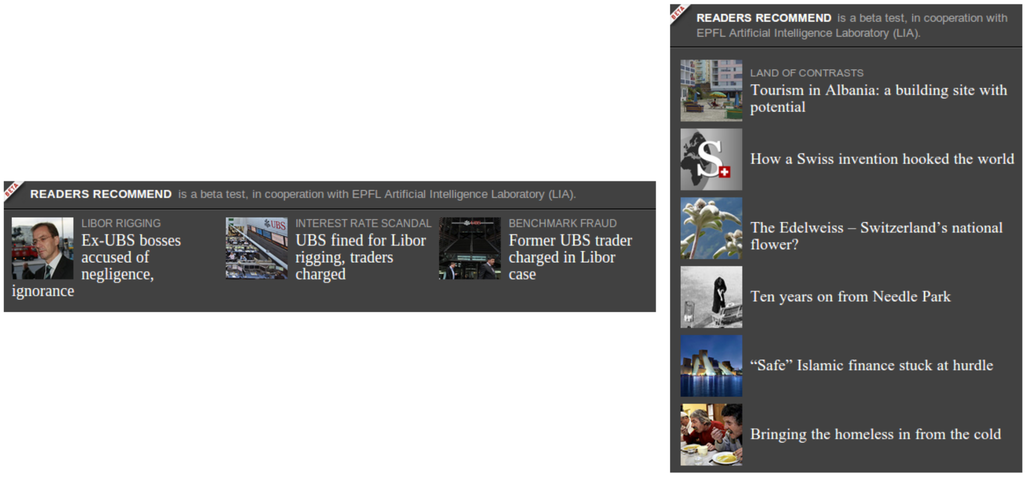
We used the offline evaluation to select the recommender systems and their parameters that could most likely generate good recommendations in a live environment. However, the difference between online and offline evaluations is striking (Figure 3). In the offline setting, it is difficult to do better than recommending most popular articles because the recommendations do not directly influence the users. This method mimics the best the behaviour of the readers when no recommender systems are in place because items on the front page attract the most clicks. In the live evaluation, the popularity-based strategy is clearly not the most interesting because users do not want to read articles they have already seen on the front page.

Some recommender systems could have been ruled out of the live trial due to their poor performance during the offline evaluation. However, the conclusion would not have been the same in a live setting [19]. Almost all research in the recommender systems literature is based on offline evaluation with the assumption that it would at least reflect the relative performance of different techniques. However, at least in this case this turns out to be wrong.

The next step would be to study the relationship of recommendation performance to other criteria such as diversity and novelty. We define novelty as the fraction of recommended articles that are not among the most popular items. The CT recommender used here achieves a novelty of about 50% [7], whereas the random strategy is at almost 100% and the most popular strategy is at 0%. Thus, a combination of offline accuracy and novelty can predict the true relative performance of the different strategies from offline data. As the system runs longer, we plan to gather more data with different recommendation strategies to firm up what this relation may be.

5.2 Click-through Rate

Although there exist many metrics [9, 21] to assess the performance of a recommender system, the click-through rate is a de-facto standard in the industry because it is often correlated to the revenues generated by the news website. Indeed, these revenues come from either advertisements (ads) displayed on the website or paid articles (or sometimes both). In general for online advertisement, there are two revenue models: pay per impression or pay per click. With the former, the news website receives monetary compensation for displaying ads while with the latter every time a user clicks on the ad. In all cases, news websites are incentivized in increasing page views. So one way to justify the perfor-



READERS RECOMMEND is a beta test, in cooperation with EPFL Artificial Intelligence Laboratory (LIA).

LIBOR RIGGING
Ex-UBS bosses accused of negligence, ignorance

INTEREST RATE SCANDAL
UBS fined for Libor rigging, traders charged

BENCHMARK FRAUD
Former UBS trader charged in Libor case

(a) bottom (stories and tickers)

READERS RECOMMEND is a beta test, in cooperation with EPFL Artificial Intelligence Laboratory (LIA).

LAND OF CONTRASTS
Tourism in Albania: a building site with potential

How a Swiss invention hooked the world

The Edelweiss – Switzerland's national flower?

Ten years on from Needle Park

"Safe" Islamic finance stuck at hurdle

Bringing the homeless in from the cold

(b) top right (only tickers)

Figure 8: swissinfo.ch's dynamic recommendation boxes: bottom and top-right position. Note that the number of recommendations is not the same.

mance of a recommender system is through its generated revenue, or click-through rate.

Zheng et al. [23] suggest that CTR might not be the optimal metric for online evaluation, because some of the clicks are for popular items that people would have chosen anyway. We have observed the same phenomenon here: recommending popular items not only results in a lower than expected CTR, but an even lower increase in the number of reads.

We believe that rather than CTR, it would be better to measure the actual difference in success@k rates between offline and online evaluations. This measure would unequivocally indicate the net effect of using the recommender system on website performance, although it may be more complex to implement as a business model.

5.3 Page Layout

A recommender system is a small piece of a bigger and complex environment. In our case, we have little to no control over the other elements of the environment in particular the placement of the recommendations on the page and possible interference with manually-generated recommendations that were also shown on the same page in a more prominent position and may overlap with the ones generated by the recommender systems.

The layout of the page is extremely important [16, 4]. On swissinfo.ch, most of the news stories are long and do not fit on the displayed area of the screen. Thus, the placement of recommendations on the page plays an important role in drawing users' attention. When recommendations are placed at the bottom of the article, users need to scroll down the page to see the recommendation box (Figure 8), and we believe that users tend not to read articles down to the end.

We were able to conduct a simple A/B evaluation regarding the placement of the recommendations on one part of the swissinfo.ch website. We compared the performance with the same recommendations but placed in different positions on the page: top right and bottom. The top-right version allows us to display more recommendations (6 stories) than the bottom version (3 stories, see Figure 8). Figure 9 illustrates the difference in click-through rate between the two placements. When the recommendations are at the top-right corner of the page, the CTR is more than double (2.25) the one at the bottom. Hence, displaying recommendations at the top-right is significantly (χ^2-test with $p < 0.01$) better than at the bottom. The placement might not be the only reason of this improvement. The width of the page limits the bottom recommendation box to only three items, while the top-right version contains 6 recommendations. So the larger choice available to the user might also play a role. An easy way to verify this claim would be to place the bottom version at the top of the page. Incorporating aspects of the page layout into the recommender algorithm could also improve the performance [13].

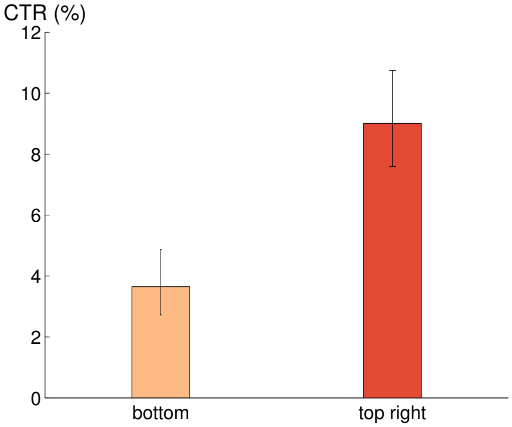


Figure 9: Online *actual* CTR on tickers for the two different placements, with confidence intervals at 95%.

Another issue that has a major impact on performance are other recommendations generated by journalists and placed on the same page. They take into account only the current article, but not the history and preferences of the reader. For technical reasons, we have no access to these recommendations and they may thus overlap with the ones generated automatically. As they are placed in a more prominent position on the page, they would draw clicks away from our recommendations. Thus, our evaluation might underestimate the correct click-through rate because of this bias.

6. CONCLUSION

We implemented a novel recommendation technique based on context trees [7, 8] on the news website *swissinfo.ch*. In a live evaluation, we demonstrated that context-tree recommender systems significantly improved the click-through rate by up to 35% and the visit length by a factor of 2.5, with the best performance for users with long enough visits to reveal their preferences. The absolute performance is somewhat impaired by the poor placement of the recommendation and lack of coordination with manual recommendations on the same site, factors which we could not influence.

Besides the confirmation of the actual performance, an important message for recommender systems research is that offline evaluations of accuracy are not always meaningful for predicting the relative performance of different techniques. In our experience, the ranking of techniques in a live environment is the inverse of what it is in the offline evaluation. Thus, better models incorporating novelty are needed.

A second important lesson concerns the value of the CTR as a measure of performance. As in some earlier studies [23], we showed that the CTR overestimates the actual impact for popular items, and thus gives a skewed impression of the actual performance. Only comparing the actual article views can give a true picture of actual performance.

Finally, we observed that the placement of recommendations is important for the click-through rate and should be carefully selected when deploying recommendations on a live website. Placing the recommendations at the top of the page more than doubled the CTR than at the bottom.

With the live evaluation, we plan to refine the users' profile by tracing users across multiple visits, and enhance their profile with socio-demographic data. We believe that algorithms will benefit by incorporating information from the manual recommendations as well. By doing so, we would avoid duplicated recommendations in both manual and dynamic lists. We intend to blend the recommendations into the page design and change their positions to a more visible place. Finally, an interesting future direction would be to consider not only news articles, but also video and audio.

7. REFERENCES

[1] J. Ahn, P. Brusilovsky, J. Grady, and D. He. Open user profiles for adaptive news systems: help or harm? In *Conf. on WWW*, pages 11–20, 2007.

[2] D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *Conf. on User Modeling*, pages 99–108, 1999.

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

[4] L. Chen and P. Pu. Eye-tracking study of user behavior in recommender interfaces. In *User Modeling, Adaptation, and Personalization*, pages 375–380, 2010.

[5] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Conf. on Rec. Systems*, pages 39–46, 2010.

[6] A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Conf. on WWW*, pages 271–280, 2007.

[7] F. Garcin, C. Dimitrakakis, and B. Faltings. Personalized news recommendation with context trees. In *Conf. on Rec. Systems*, pages 105–112, 2013.

[8] F. Garcin and B. Faltings. Pen recsys: A personalized news recommender systems framework. In *Conf. on Rec. Systems*, pages 469–470, 2013.

[9] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *Trans. on Info. Systems*, 22:5–53, 2004.

[10] W. IJntema, F. Goossen, F. Frasincar, and F. Hogenboom. Ontology-based news recommendation. In *Workshop on Data Semantics*, page 16, 2010.

[11] B. Kille, F. Hopfgartner, T. Brodt, and T. Heintz. The plista dataset. In *News Recommender Systems Workshop*, pages 16–23, 2013.

[12] E. Kirshenbaum, G. Forman, and M. Dugan. A live comparison of methods for personalized article recommendation at forbes.com. In *ECML - PKDD*, pages 51–66, 2012.

[13] K. Lerman and T. Hogg. Using a model of social dynamics to predict popularity of news. In *Conf. on WWW*, pages 621–630, 2010.

[14] L. Li, W. Chu, J. Langford, and R. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Conf. on WWW*, pages 661–670, 2010.

[15] J. Liu, P. Dolan, and E. Pedersen. Personalized news recommendation based on click behavior. In *Proc. of IUI*, pages 31–40, 2010.

[16] P. Pu, L. Chen, and R. Hu. A user-centric evaluation framework for recommender systems. In *Conf. on Rec. Systems*, pages 157–164, 2011.

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW*, pages 175–186, 1994.

[18] A. Said, A. Bellogin, J. Lin, and A. de Vries. Do recommendations matter?: News recommendation in real life. In *CSCW*, pages 237–240, 2014.

[19] A. Said, B. Fields, B. Jain, and S. Albayrak. User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm. In *CSCW*, pages 1399–1408, 2013.

[20] A. Said, J. Lin, A. Bellogin, and A. de Vries. A month in the life of a production news recommender system. In *Workshop on Living Labs for Information Retrieval Evaluation*, pages 7–10, 2013.

[21] G. Shani and A. Gunawardana. Evaluating recommendation systems. In *Recommender Systems Handbook*, pages 257–297. Springer, 2011.

[22] M. Tavakolifard, J. Gulla, K. Almeroth, F. Hopfgartner, B. Kille, T. Plumbaum, A. Lommatzsch, T. Brodt, A. Bucko, and T. Heintz. Workshop and challenge on news recommender systems. In *Conf. on Rec. Systems*, pages 481–482, 2013.

[23] H. Zheng, D. Wang, Q. Zhang, H. Li, and T. Yang. Do clicks measure recommendation relevancy?: an empirical user study. In *Conf. on Rec. Systems*, pages 249–252, 2010.