Applying Multi-Armed Bandit on top of content similarity recommendation engine

Andraž Hribernik
Zemanta
andraz.hribernik@gmail.com

Lorand Dali
Zemanta
lorand.dali@zemanta.com

Dušan Omerčević
Zemanta
dusan.omercevic@zemanta.com

Dejan Lavbič
University of Ljubljana
dejan.lavbic@fri.uni-lj.si

ABSTRACT

One of the goals of every news publisher is to prolong users' browsing sessions by convincing them to read additional content besides the initially requested. Consequently, most news publishers use recommendation engines to keep users engaged by recommending them content which they might like to read next. In this paper we present how to improve an existing news recommendation system deployed on the mobile news site of one of Zemanta's partners. With this publisher the users are browsing news sequentially by scrolling through a continuous stream of items on their mobile devices. Our goal is to order the recommended items in a way to increase probability that the users will click one them and read them. The existing system orders the recommendations by similarity to the article that the user is currently reading. We build on top of the content-similarity approach by also estimating the probability that the recommended item will be clicked. We use click-through data, and take a Bayesian approach to estimate the probability of a click; moreover, we propose the use of hierarchical models as a way of including data from contextually similar sessions into the estimation. The recommendations are performed using a multi-armed bandit approach. We try to find a trade-off between "exploitation" of items likely to perform well and "exploration" of items where not much data is available. When evaluated on data collected from a mobile news publisher, the proposed methods showed significant improvements compared to the existing system.

Categories and Subject Descriptors

G.3 [**Mathematics of Computing**]: Probability and Statistics; I.2.6 [**Computing Methodologies**]: Learning

General Terms

Algorithms, Experimentation

Keywords

exploration vs. exploitation, recommendation systems, mobile news

1. INTRODUCTION

Nowadays, there are more and more users accessing web sites via mobile devices. Consequently, news sites have been adapted for mobile web browsers and their recommendation systems have changed as well. Mobile web sites are customised in a scroll-down manner and content is displayed in vertical order (Fig 1). In our case, when user reaches the bottom of the displayed article, the first recommended item (news label) is displayed. While the user scrolls down, new recommendations are displayed continuously. If the user is interested in the recommended news, they click on them. All recommendations and clicks are reported to a back-end system, which stores information about the context of every recommendation. This provides the information under which article the recommendation or impression has been shown.

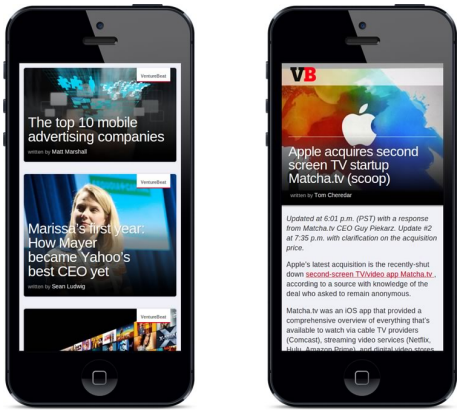

Figure 1: A snapshot of Venturebeat mobile web site using Zemanta Streams.

This paper discusses a concrete engineering problem of improving a content-based recommendation engine with two specific settings. The first is a continuous recommendation stream, in which the recommended items are ordered by content similarity w.r.t. the currently shown article. Secondly, the recommendation engine has access to statistics about clicks. In contrast, statistics about users and what they like or dislike is not available in our case. Therefore, content filtering [5] based recommendation engines are not a suitable choice for this problem.

The existing Zemanta's recommendation engine [1] makes recommendations based on content similarity. The objective is to place items that are most likely to be clicked on as high as possible. Therefore, we would upgrade the existing system using a Multi-Armed Bandit (MAB) on top of content similarity recommendation. Items that are similar to the currently read article, but with a higher Click-Through Rate (CTR), would be positioned in better positions.

We reviewed some existing approaches in section 2. The main part of this paper (section 3) will discuss several methods for estimating CTR or recommended items. This is used by MAB algorithm. The results are reported in section 4. The conclusions and guidelines for future research are reported in section 5.

2. RELATED WORK

The most common approaches for dealing with recommendations are content-based [8, 11] or collaborative filtering [5, 3] recommendation systems. Advanced applications of content-based recommendation systems [8] take into account the user's past actions. while collaborative filtering is based on user's past actions. In contrast, many advertising systems cannot access data about users and their preferences. The goal of these systems is to maximise profit. A common approach for dealing with that kind of problem is a Multi-Armed Bandit (MAB) [2], since its tecniques exploit the most profitable ads and explore potentially better ones. Some advertising systems based on MAB have been updated in a way that they take into account the context of the clicked ads or recommendations [13, 6]. Authors in [6] were working on solving the problem about selecting top Yahoo news. This is a problem very similar to ours, but they had access to data about users and their preferences, which we don't. The cited contextual MAB approaches used ϵ-Greedy and Upper Confidence Bounds MAB techniques. This paper will discuss a contextual MAB approach, which uses the Randomised Probability Matching (RPM) approach [12] for trade-off between "exploration" and "exploitation".

3. METHODS

MAB approaches optimise trade-off between exploiting the most profitable items and exploring potentially better ones. We applied the RPM [12] MAB technique on top of the existing content-based recommendation engine. RPM estimates the items' CTR as a random value distributed among beta distribution. Beta distribution needs two parameters: α and β. We could directly transform the number of clicks of an item to the value of α. Similarly, β is the number of shows subtracted by the number of clicks of a particular item. The existing recommendation system provides a list of items most similar by content. Items are ordered by the extent of similarity. Our goal is to reorder similar items in a way that the most likely clicked items would be positioned in top positions. We order similar items decreasingly with regards to an estimated CTR retrieved by RPM. In most cases the order would be the same as if we ordered items by their CTR. Sometimes, the order of items depends on randomness, which is the essential idea of MAB. We tried several different methods for estimating the parameters α and β and those will be discussed in the following sections. Performances of the following methods are reported in section 4.

3.1 Adjusted CTR

A click of a recommended item is most likely to happen if the item is displayed in the first position. If a click occurs in any other position, this click is more important than the click of an item in the first position. Due to this we computed a position factor (w_p), which was based on average CTR per position. The position factor was used as a weight factor for computing parameter α. Parameter α was still computed as a sum of all clicks, but every click was weighted by the position factor. This approach improved performances and we later used the adjusted CTR on all methods below.

$$w_p = \frac{\text{CTR}_1}{\text{CTR}_p},\ \alpha = \sum_{p=1}^{\text{max pos.}} w_p \cdot \#\text{clicks}_p \qquad (1)$$

3.2 Global CTR

Ordering of the recommended articles was done using a global CTR; this means that the context of a stream was ignored or, in other words, the article under which the item was shown was not observed. Parameter α was computed as a sum of all adjusted clicks that occurred for an article we would like to recommend and β was the number of all shows subtracted by the number of clicks.

3.3 Stream CTR

In contrast to the previous method, Stream CTR observed the behaviour of recommended items only in context of the current stream (i.e. the currently read article with its corresponding recommendations). This means that α is the adjusted number of clicks of the recommended article under a particular article at the time of recommendation. Similarly, β is the number of shows subtracted by the number of clicks of the current stream.

3.4 Top 100

The intuition behind this method is that we should not make a recommendation based on data with high bias, since recommended items were seldomly shown many times, especially in a context of stream. Due to this expected CTR has very high variance. This method tries to estimate the items' CTR using data about the recommended items from similar streams. This approach is an extension of the Stream CTR method. It does not estimate CTR with data from one stream only, but collects data from streams most similar to the current stream as well.

Parameters α and β are computed by taking into account the hundred streams most similar (content-wise) to the current stream of shown article. The value of α is the sum of all clicks of the recommended item in most similar streams and β is the sum of all shows subtracted by the value of α.

When evaluating this method we used different numbers of similar streams. We achieved the best results for estimating parameters α and β with twenty most similar streams.

3.5 Estimating initial parameters α and β

Parameter values of α and β must be positive. The most simple and basic initialisation of parameters could set both values to one. In that case the estimated value of CTR using

RPM would be uniformly distributed on interval (0,1). We used different settings of initial parameters so far, but all of them were set experimentally. The purpose of this method is to estimate the initial values of α and β with the known CTR of items.

We tried to find the most probable parameters α and β w.r.t. the CTR distributions of the known items $(X_1, X_2, ..., X_n)$. The following equation 2 applies Bayes theorem.

$$P(\alpha, \beta | X_1, X_2, ..., X_n) = \frac{P(X_1, X_2, ..., X_n | \alpha, \beta) P(\alpha, \beta)}{P(X_1, X_2, ..., X_n)} \quad (2)$$

Denominator in equation 2 is constant, therefore we could ignore it in our simplification.

$$P(\alpha, \beta | X_1, X_2, ..., X_n) \approx P(X_1 | \alpha, \beta) ... P(X_n | \alpha, \beta) P(\alpha, \beta) \quad (3)$$

We tried to maximise equation 3. The rejection sampling [10] approach was used for obtaining parameters α and β.

The technique described above was applied in two different settings. Firstly, the initial parameters α_0 and β_0 were estimated for the entire dataset of items. This means that the values of $(X_1, X_2, ..., X_n)$ were global CTRs of all articles on the day before the evaluation started. Concretely, when the recommendation was made for one item, the initial values of α_0 and β_0 were added to estimated α and β from the item CTR. The second application of this method computed the initial values of α_0 and β_0 for every recommendation separately. Content similarity recommendations system provides a list of items most similar to the currently read article. The values $(X_1, X_2, ..., X_n)$ were represented as stream CTRs of similar items. These values $(X_1, X_2, ..., X_n)$ were used to estimate initial values α_0 and β_0 for every recommendation. To order items, the initial parameters and the stream CTR values of similar items were used by the RPM approach

3.6 Greedy method

This method was implemented and evaluated as a reference method. The Greedy method orders items with regards to their global CTR; a randomised step, which was used by the MAB approach, was skipped.

4. EVALUATION AND RESULTS

All our methods were evaluated offline on a real dataset. We had access to three months traffic from Zemanta Streams recommendation engine. The last 10-days recommendations were used for evaluation. Other data was used as pre knowledge for methods described in section 3. The performances of the recommendations in that period were known. The evaluation technique that was used is described in [4]. Most of the clicks occur in top positions, especially in the first one. Users normally trust the existing recommendation system and only check top items. If a click happens in a position further down, this article is likely to be much more interesting for the user than all articles recommended before the clicked one [4]. Therefore, the clicked article should be positioned in the first place, because it was apparently more appealing and interesting than the others. The perfect recommendation would be that all clicked recommendations would occur in the first position.

We measured several different metrics. The most informative were the average position [7] improvement and the number of clicked articles recommended in the first position (Precision@1) [9]. The average position improvement compares the positions of original clicked items with the position recommended by our methods. This improvement was measured for each individual recommendation from our 10-day dataset. Average improvement was computed at the end of each evaluation and is reported in Table 1. The second metric shows how many clicked items were recommended in the first position. The report does not include the exact number of items placed in the first position, but a relative number regarding a perfect recommendation (RPR).

$$RPR = \frac{\text{\# first placed recommendations}}{\text{\# first placed recommendations in perfect scenario}} \quad (4)$$

Table 1: Results of relative number of clicked items recommended on first position and average position improvement per method

Method	# first position[%]	avg. pos. improvement
Stream CTR	47.0	4.09
Top 100 (20)	46.4	4.02
Estimating initial parameters α & β	46.5	3.85
Global CTR	41.9	3.6
Greedy	41.8	3.65
Existing Zemanta System	22.4	/

The results in Table 1 1 show that three methods performed much better than the others and depict the Stream CTR method as likely to be the most promising one. Using this method 47% of clicked items were positioned on top and the average-clicked item was positioned more than 4 positions higher than the items in the original recommendations. Similar performance was achieved by the Top 100 method and the method which estimates parameters α and β for each recommendation separately.

The best performances were achieved using the Stream CTR method, because this method is most sensitive to any changes in the items' CTR. This is very important especially at the beginning of the stream, when information about the CTRs is limited. This immediate adjustment to changes of CTR is seen on Fig 2. When enough data is available, all methods perform similarly. It is, however, possible that with somewhat different settings of the recommendation engine we would also be able to achieve better results using the Top 100 most similar method or the Estimated α and β for every recommendation.

Fig 2 shows which item would be placed in the first position if we selected it using method Stream CTR. We observed the article with the title "Yahoo's mobile video streaming app is now on Android too". The existing system was content-based, therefore, the recommended items were ranked by similarity and always in the same position. The item in the first position was titled "Yahoo may build its own YouTube" and its final CTR was 1.5%. The item in the forth position was titled "Pandora finally comes to Chromecast via Android & iPhone apps" and its final CTR was 2%. The fifth

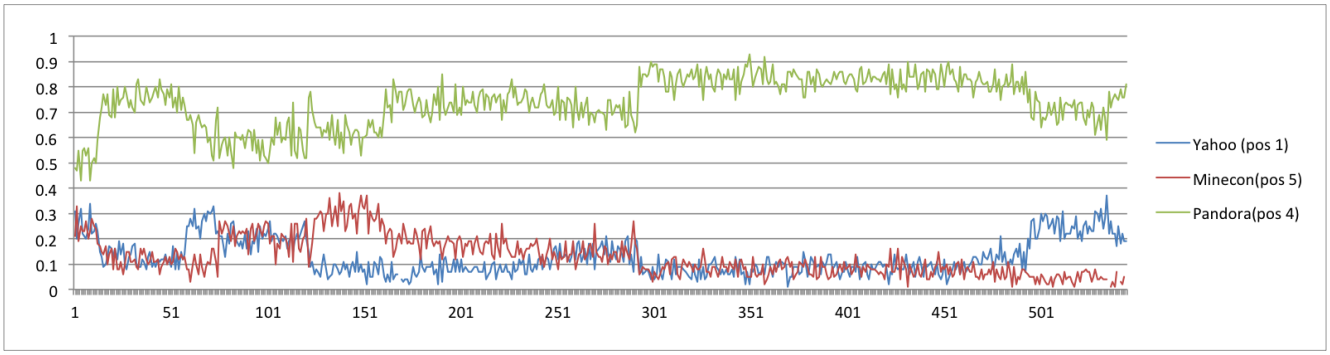

Figure 2: Probability of items being positioned on the first position using Stream CTR. This simulation was made for stream with article Yahoo's mobile video streaming app is now on Android too.

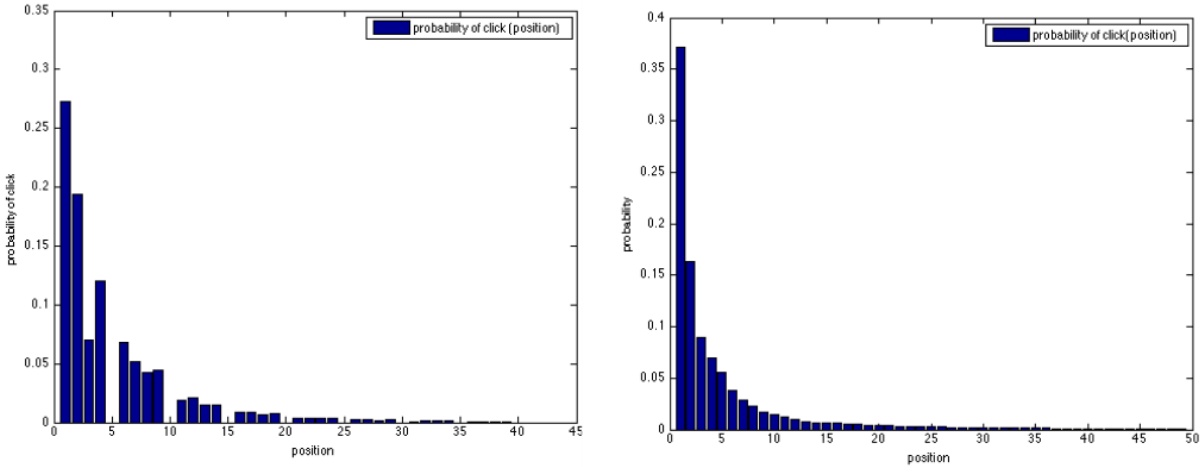

Figure 3: These charts show a distribution of clicks regarding the first fifty positions. The left image is click distri- bution of the existing Zemanta's recommendation engine based on content similarity. The image on the right shows click distribution of our best approach. The curve of the improved recommendation engine is much steeper, which proves the improvement of the existing content similarity approach.

position item was "Minecon live: Watch Mojang's Minecraft event right here" and its CTR was 0.8%. X-axis shows the sequence number of recommendation, while Y-axis shows the probability of an article being shown in the first position using MAB. The article with the highest CTR and therefore the one most likely to be clicked on, would be recommended in the first position through the entire stream. Surprisingly, the most promising item was identified after fifteen recommendations. Other methods (Global CTR, Top 100, etc) identified the most promising items after more recommendation and this is the main difference in performances between them.

Fig 3 shows that distribution of the clicks w.r.t the position is much steeper using the Stream CTR method than the existing content recommendation engine. The evaluation was made offine, and therefore we cannot directly report improvement of the CTR. Using this visualisation we show that the updated system would perform better, since the items with more clicks were recommended in top positions. This would consequently lead to a higher CTR as well. Compared to the Greedy method, the Stream CTR method percentage of improvement (with regards to the number of items clicked in the first position) was 7% higher. Nonetheless, the Greedy method improved the existing Zemanta's recommendation engine as well.

5. CONCLUSION

In this paper we demonstrated that applying MAB on top of the existing content-based recommendation is a good choice. The performances have improved significantly. Many different methods for estimating the items' CTR were evaluated. Three methods performed better than the rest. The best results were achieved by the Stream CTR method. The most beneficial is that this method is very simple for implementation and much less complex from a computational perspective than the other two promising methods. Most improvement of the existing recommendation engine was achieved by observing the context of a click (Stream CTR) and using a position factor for clicks that occur in higher positions (Adjusted CTR).

Future research will focus on finding better settings of the initial parameters α and β and trying to find a hybrid method, which would take into account multiple methods described in this paper.

6. ACKNOWLEDGMENTS

We thank Zemanta for providing us a data resources for the research.

7. REFERENCES

[1] www.zemanta.com, June 2014.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[3] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):81–173, 2011.

[4] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

[5] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[6] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[7] B. Libby. Modeling clickthrough probabilities. 2009.

[8] P. Lops, M. De Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[9] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[10] R. M. Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003.

[11] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[12] S. L. Scott. A modern bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010.

[13] J. L. T. Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. 2007.