Recommending Relevant Applications for Users of SMS based Platforms

Gopi Krishnan Nambiar, Samit Paul

Intuit Technology Services Private Limited, Campus 4A, Ecospace Tech Park, Bangalore 560103, Karnataka, India
{Gopi_Nambiar, Samit_Paul}@intuit.com

Abstract SMS based platforms offer many apps (applications) for the user to choose from. Therefore, it is of paramount importance to surface the most relevant apps to a user in order to increase their engagement and improve discoverability. A data driven approach for the recommendation of relevant applications to these users is proposed in this paper. Experiments were conducted on the users of a popular SMS platform in India 'txtWeb'. The result metrics were closely tracked through an A/B test that carried out unbiased experiments on live txtWeb traffic. Significant improvements were witnessed in the average number of apps accessed per user after the introduction of this new algorithm.

1 Introduction

Short Messaging Service (SMS) is one of the most widely used features of a mobile phone, especially in emerging market economies like India, where a considerable majority of the mobile phones are low cost devices with very minimalistic feature sets and low processing power. Due to the high cost of the data plans (mobile internet packs for internet enabled phones), SMS based platforms providing information services have become popular in emerging markets. These enable users to access various types of mobile content, e.g. local directories, relevant offers, sports updates etc. The SMS based platform ecosystem involves two key players: developers and users. The developers build various apps (applications) which deliver dynamic content listed above (sports updates, offers) and the users consume this content, since this content is otherwise unavailable offline. An app on an SMS based platform is similar to a domain name on the Internet. It is a unique keyword which a user can text to the SMS based platform and the platform returns the relevant content for that keyword. This procedure is analogous to typing a domain name on the web browser and the website opening up, only in this case a user would receive a text message as the response.

Kamvar et al. [1] found that the mobile search click-through rate and the search page views per query were both significantly lower in comparison to desktop search. The study also found the persistence of low-end mobile phone (feature phone) users was very low indicating that the vast majority of users for mobile search submit queries with a specific topic in mind and their search often does not lead to exploration. A

second study by Kamvar et al. [2] showed that the diversity of search topics for low-end phone users was much less than that of desktop or iPhone based search. Considering the limited real estate and cost involved to send messages over the SMS medium, it becomes imperative for SMS based platforms to recommend the most relevant and interesting apps to an end user. This also becomes a means of engaging existing users by recommending relevant apps.

Over the past decade recommender systems have gained significant traction due to the exponential growth of world-wide-web and emergence of e-commerce [4]. Recommender system is an information filtering technology to cater set of N relevant items that will be of interest to a given user (known as top-N recommendation problem) or to predict a user for a specific product (prediction problem). Some of the notable applications of recommender systems are witnessed in product suggestions for e-commerce websites (e.g. Amazon, eBay), movie suggestions (e.g. NetFlix, IMDB), music recommendations (e.g. Pandora, iTunes Genius) and people suggestions in social network (e.g. Facebook, LinkedIn). A detailed review and classifications of recommender systems can be found in Park et al [3]. In this paper, we describe how a data driven app recommendation engine helped in improving the engagement rate of the users of an SMS based platform via providing top-N recommendations.

2 txtWeb – an SMS based Platform

txtWeb (http://www.txtweb.com) is a highly efficient real-time communication channel for textual, time-sensitive, non-intrusive, relevant and short updates. It is a global platform where anyone with a mobile phone can discover and consume information and content just by 'texting' keywords, and receive back content. txtWeb aims to provide information accessibility to those population segments that lack Internet access by exploiting the universality of text messaging services. It is a popular SMS based service in India, with the total number of users crossing 7 million (2 million active users per month). There are numerous apps on txtWeb contributed by developers. The end user can choose to use any app of his choice by texting a specific keyword, which acts as a unique identifier.

In txtWeb, a recommendation is a small snippet of text inserted at the end of every response message. This is the primary means of app discovery for a txtWeb user. It contains a short description about a particular app and its usage instructions. Currently, the sole method of recommendation on txtWeb is by means of a 'Manual Recommendation', which is served in a cyclic fashion to an end user based on a predefined list of apps. For instance, if the predefined list contains apps K_1, …, K_n, then for the first time access of a given user, the app K_1 will be recommended, app K_n will be recommended for the n^{th} request and this process goes back to recommend K_1 for the $(n+1)^{th}$ request. Due to the manual nature, this process is not scalable and adaptive. Also this does not serve relevant apps to the user based on his current accessed app. In order to circumvent these shortfalls of manual recommendations, we propose an automated data driven app recommendation engine.

3 Methodology

The objective of the app recommendation engine is to recommend a list of most similar N apps for a given app, which is accessed on the txtWeb platform. Let R be an $m \times n$ user-app collaboration matrix containing historical request count of m users for n apps. In this matrix r_{ij} represents the count of request of i^{th} user for the j^{th} app. Then we normalize the raw entries of matrix R with respect to each user to bring the values in the same scale so that we can compare across the entire user-base. The normalized matrix is defined as $R' = \left[r'_{ij} \right]$, where $r'_{ij} = r_{ij} \Big/ \sum_j r_{ij}$. Then this normalized user-app collaboration matrix is used to calculate the pair-wise cosine similarity between apps as defined in equation **(1)**.

$$Sim(K_l, K_m) = \frac{\overrightarrow{K_l} \bullet \overrightarrow{K_m}}{\left\| \overrightarrow{K_l} \right\|_2 \left\| \overrightarrow{K_m} \right\|_2} \qquad (1)$$

For example, for app K_1 and K_2, the similarity $Sim(K_1, K_2)$ is the cosine distance between vectors $(R_{11}, R_{21}, \ldots, R_{m1})$ and $(R_{12}, R_{22}, \ldots, R_{m2})$. In this manner we can compute the similarity scores between all possible pairs of apps. Therefore for a given app K_l we can sort the set of other apps (i.e. $\{Sim(K_l, K_j)\}_{j \neq l}$) in a descending manner by their similarity score with K_l in order to recommend top-N most similar apps.

4 Experiment and Results

The important steps of the app recommendation engine have been highlighted in Figure **1**. Initially, the raw request log from txtWeb platform is processed as part of the pre-processing module. This consists of the following rules:

1. Consider only those apps with distinct user count greater than 200.
2. Consider only those apps with bounce rate is less than 70% (Bounce rate represents the percentage of users who use the platform once and then never come back).
3. Consider only those apps, which are published and active.

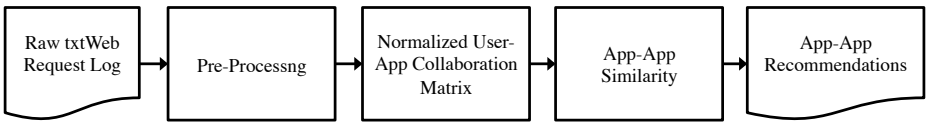

Figure 1: Key Steps of App Recommendation Engine

Next, the normalized user-app collaboration matrix is constructed from the pre-processed txtWeb request log. Finally the all pair app-app similarity scores are computed to generate the list of app-app recommendations. In order to create the list

of recommendations, we have considered three months request log data from November 18, 2012 to February 16, 2013. For this period we had started with 139,551,282 SMS requests for 2,497,058 users and 5,029 apps. After the pre-processing module filters were applied, 132,410,088 SMS requests remained for 2,392,477 users and 383 apps. The final app-app recommendations were computed based on this data. Given the high volume of requests and users, we implemented this pipeline in Hadoop platform.

The experiment was conducted as an A/B test for txtWeb users during the period of March 14, 2013 to April 2, 2013. The test bucketing logic was based on the 8th digit of the user's mobile number. If the digit was even, the user received recommendations from the data driven engine and if it was odd, then the user received the existing manual recommendation. The bucketing logic was chosen in the above manner, since it was observed that the spread of the users in the test set (50.31%) and the control set (49.69%) were almost the same.

The following result metrics were tracked for deciding the success or failure of the test:

Average apps-per-user: This metric is a direct indicator of user engagement. It is calculated by dividing the total number of apps accessed by the total number of users exposed to the test for the given test duration. This is a good measure of the test result as it helps understand if the data driven recommendations resulted in users actually discovering and using new applications on the txtWeb Platform.

The primary hypothesis of the experiment was that the average number of apps accessed per user would increase as a result of the data driven recommendation that are being served to the user.

To ensure that we did not start with any initial bias, we measured the apps-per-user for the test and control sets before the test duration. The results are shown in Table 1. It should be observed that the average apps-per-user for the test set (2.45) and the control set (2.47) differed by only 0.8%, and also that the control set was doing better in this metric as compared to the test set.

After the data driven recommendation test was performed, we observed that the average apps-per-user for the test set was 3.05 whereas for the control set it was at 2.68, as shown in **Table 2**. There was an improvement of 13.8% in the average apps-per-user metric for the data driven recommendation test set as compared to the control set.

Table 1. Average apps-per-user metric for test and control set **before** the test

User set	Avg. apps-per-user
Test set	2.45
Control set	2.47

Table 2. Average apps-per-user metric for test and control set **during** the test duration

User set	Avg. apps-per-user
Test set	3.05
Control set	2.68

5 Conclusion and Next Steps

We proposed a data driven recommendation system in the above paper and tested our hypothesis through an A/B test, which compared this system with the existing manual recommendation system on the txtWeb Platform. This change led to a 13.8% improvement in the average number of apps accessed per user, which is a significant improvement in user engagement rate. This is a clear indication that the data driven recommendation approach is the desired approach for improving app discovery for end users of SMS based platforms.

Currently, the recommendation is based on an app being used by any user of the platform and does not take into consideration the personalization aspect. Therefore, the next step is to adopt techniques like collaborative filtering to cater more personalized recommendations at a user level. In order to make the recommendation system more realistic and robust, we can also incorporate information about trending apps and few expert suggested popular apps.

References

1. Kamvar, M., Baluja, S. A large-scale study of wireless search behavior: Google mobile search. Proc. SIGCHI conference on Human Factors in computing systems, 701–709, 2006.

2. Kamvar, M., Kellar, M., Patel, R., and Xu, Y. Computers and iphones and mobile phones, oh my!: a logs-based comparison of search users on different devices. Proc. 18th International Conference on World Wide Web, 801–810, 2009.

3. Park, D.H., Kim, H.K., Young Choi, I.Y., Kim, J.K., A literature review and classification of recommender systems research. Expert Systems with Applications, 39(11): 10059-10072, 2012.

4. Resnick, P., and Varian, H.R. Recommender Systems. Communications of the ACM, 40(3): 56-58, 1997.