Bruce Meeks Jr

Fall 2015 Machine Learning CSE 6740

December 7th, 2015

**Hawkes Process Mobile App Demo Report**

I have published a prototype Android mobile app to the Google Play Store that demonstrates the application of the Hawkes Temporal Point Process model to Tudiabetes forum data. In the Google Play Store app on an Android phone, search and download the following app called:

“**A Hawkes Process Demonstration**” or follow this link to Google Play:

<https://play.google.com/store/apps/details?id=eotm.indefdom.hawkes_process>

This write-up will give a brief background on the mobile app project itself, detail the preliminary set-up performed, technical implementation details, future work, and conclusions.

**Background**

In Fall 2014 another student and myself as a CSE 8903 Special Problems project were assigned the task of exploring the fitting of Hawkes Temporal Point Process model to raw Tudiabetes forum data for purposes of generating recommendations to forum users. These recommendations would be forum posts that the users may be interested in based upon their temporal activity patterns on the forum. This recommendation feature can be particularly valuable to all social media platforms whose goal is to promote user participation and collaboration.

The model employs a conditional intensity function which is a convenient and intuitive way of specifying how the present depends on the past in an evolutionary point process. Here the model grows by positive parameter α and decreases exponentially towards positive parameter μ each time a new forum event is observed. To make the model as realistic as possible, however, the “Marked” version of the model was chosen to account for additional information found during the time of a forum event. Such additional information are considered *marks* during times in history of a temporal point process. See Figure 1.



**Figure 1.**

**Model Fitting to Learn Alpha Similarity Parameters**

Learned alpha similarity parameters are used to recommend the forum posts of some user *j* to some user *i*. The alpha value between two forum users represents the strength of prediction that user *i* will be interested in the forum posts authored by user *j*. How then are these alpha similarity parameters learned in order to make such predictions? These alpha entries of the two-dimensional matrix Aof forum user pairs are learned by fitting the Hawkes Process model with the raw Tudiabetes data. Particularly, the data from the “rawpost” database table. This “rawpost” database table has the following schema:

**id | rawdiscussion | parent | author | content | date | rawdate | mark**

The “rawdiscussion” column contain foreign key ids to discussion thread titles in the “rawdiscussion” table. The “parent” column contains foreign key ids of the root post that the current post content is replying and/or commenting on. It is these dynamic events (i.e. posts, comments, and replies) that increases the intensity *K* of the Hawkes Process modeland are most instrumental in fitting the Hawkes Process model to learn the alpha similarity values between all forum users. **Moreover, the alpha similarity value between two arbitrary users in row *i* and column *j*, Aij strengthens in each instance user *i* replies and/or comments on a forum post authored by user *j***.

Comments are treated as *nested* replies within a discussion thread. Note the intuitive columns names in the “rawposts” database table. The relationality of the foreign key ids allows the identification of which posts are associated with which thread and also identifying the authors of discussion titles and parent posts. Fittingly the time series component of the Hawkes Process model are fitted with the timestamps of the posts content.

**Mobile Application**

The second part of the Special Problems project was to develop a mobile app client user interface to display the recommendation results yielded by the model. Here the intuition is that with the paradigm shift to social media and mobile technology, it is only fitting to make dynamically generated accurate recommendations accessible to the fingertips of users via a mobile app.

**Preliminary Set-Up**

The raw Tudiabetes data I received was stored in a sqlite3.sql file and contained three relational database tables; namely: 1) forum, 2) rawdisscussion, and 3) rawposts, respectively.

I made it my first priority to code automation scripts (MATLAB and Java) that would migrate and import these database tables and their respective data into a remote private cloud server running PHP and mySQL. Having the raw data remotely hosted on cloud infrastructure offers scalability, portability, synchronization ability, and less storage intensive on the mobile device.

In addition to the raw Tudiabetes data set, I was also given the aforementioned mu ( µ ) and alpha ( α ) positive parameter values that were learned from the Hawkes Process after being ran on the Tudiabetes raw data input. A user.txt file was also provided in which 3,083 Tudiabetes forum user ids were listed each on a separate line as such:

**1: cmonryud567dfs95nf83…….**

**2: ssdrwcx54vcfsyuvdtddy.........**

**3: def873xbtdddj43ewodfv…….**

**………..**

**………..**

**3083: a56fe3bnsssl ……….**

I coded a script to migrate and import these user ids to the remote PHP and MySQL server as well for convenient access. Similarly, the mu.MATLAB data file contained 3,083 µ values on separate lines indexed consistently with the user ids in user.txt. The µ values represented the activity rate of each user. The The alphaBig.MATLAB data file, however, consisted of a 3,083 x 3,083

2-Dimensional matrix corresponding to 9,443,329 total alpha value entries. These alpha entries are learned parameters for each forum user-pair similarity corresponding to user i and user j at entry Aij. As expected this large recommendation matrix was extremely sparse.

**Alpha Benchmark**

The alpha values corresponds to the similarity strength between two forum users, but in order to determine if an alpha value is high (strong recommendation) or not I had to establish a relative alpha benchmark value. I computed an alpha benchmark by iterating through the entire 3,073 x 3,073 alpha entry values and for the non-zero entries found the 80th percentile of the learned alpha entries. This value was .**001563. (Note alpha can be toggled and will yield different recommendations)**

**Technical Implementation**

With the raw Tudiabetes data stored on the cloud and having an alpha similarity benchmark, I was now able to begin implementing the functionality of app that identifies and dynamically loads recommendations for each forum user id.

The app begins by explaining briefly about Hawkes Process and its potential value to social media. When the user clicks “BEGIN”, the app immediately begins making several asynchronous calls to the remote server PHP API that I coded as well to retrieve the raw Tudiabetes data and return to the mobile client. Each of the calls returns Tudiabetes user ids, raw posts, forums, and discussions from the remote database. In the mobile client code I then utilized several abstract data structures to encapsulate and cache a lot of this relational data that is returned. These abstract data structures (i.e. HashMaps, ArrayLists, Adjacency Matrices, etc) were instrumental in providing a dynamic user friendly interface for displaying and opening recommendations when queried by the client.

The most important data structure used, however, was an adjacency matrix which I construct dynamically at run-time to hold all recommendations. The adjacency matrix is an array of Array Lists where each index of the array corresponds to a user id index from user.txt. Each index will contain either an Array List of recommended user ids because Aij >= the alpha benchmark or if no recommenders were found the index will contain NULL. Because the initial 3,073 x 3,073 user pair matrix was so large, it was too computationally expensive to dynamically extract recommendations from it without compromising user experience, thus for demonstration purposes I took a sample set of the first 100 x 100 user pair entries to construct the adjacency recommendation matrix from.

Traversing through the 100 x 100 sample alpha entries of matrix A the recommendation adjacency matrix is constructed dynamically as such when Aij > .**001563** meaning user j forum posts are recommended to user i.Therefore the adjacency matrix will link users to their recommended users in which posts they may be interested in.

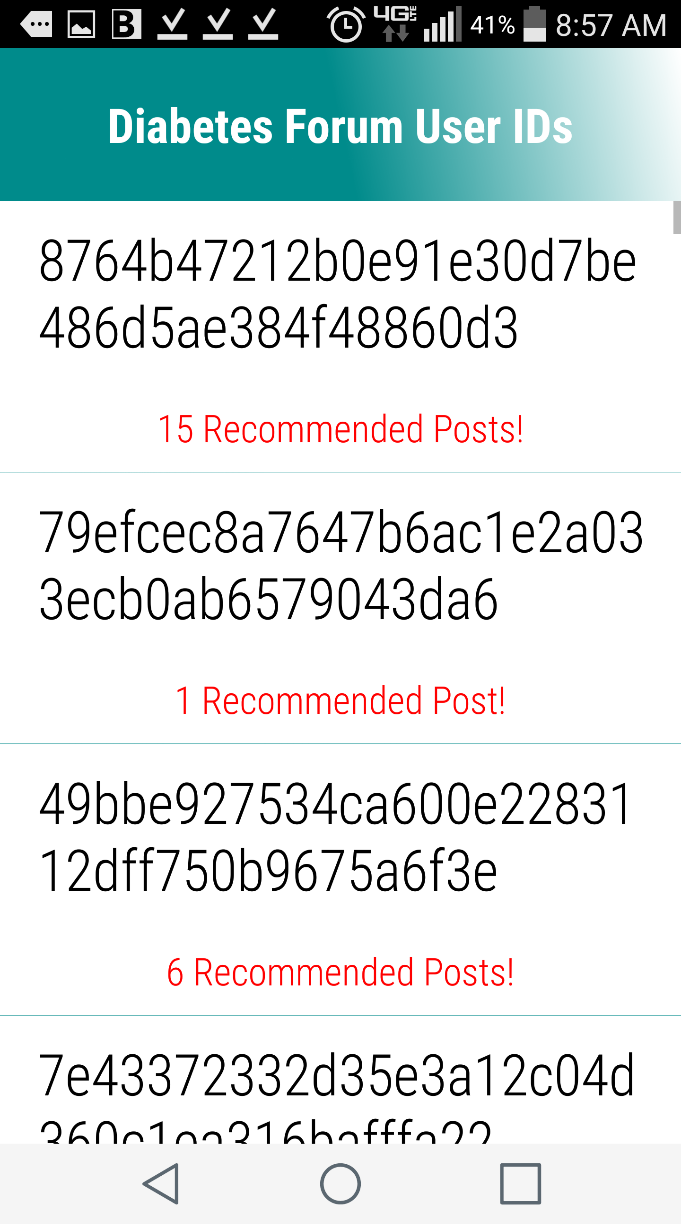
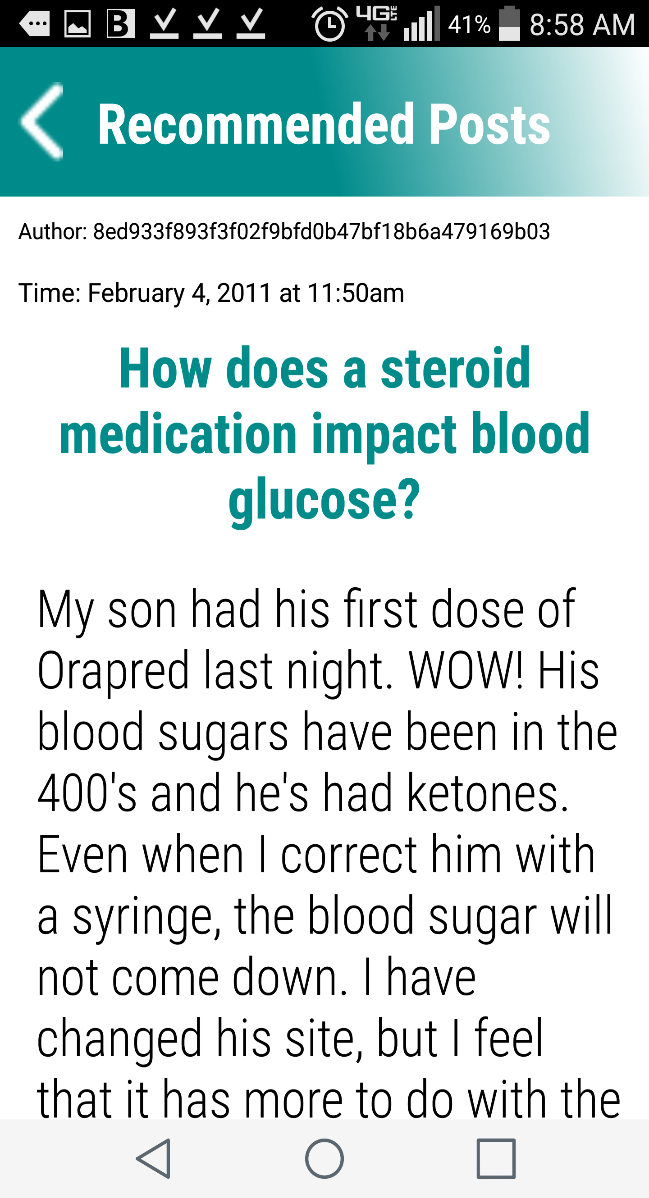
NULL

|  |
| --- |
| **userID\_1** |
| **userID\_2** |
| **…** |
| **userID\_100** |

A list view in the mobile client interface is then created containing all user ids from the database and the total number of recommendations found for each user id. This recommendation quantity appears in red and is computed by traversing all users recommended to each user at each index and then iteratively finding the total number of posts created by each of those recommended authors.

The app is very user friendly, dynamic, and responsive in that upon selecting a user id, all of the recommendations for that forum user are immediately listed with forum title, author, date/time, and the post content itself. See Figure 2 and Figure 3, respectively.

**Figure 2 Figure 3**

**Future Work**

An improvement area would be to add functionality that allows the app user to adjust the alpha benchmark and have the recommendations update dynamically. This would allow the user to adjust similarity strictness of the alpha percentile to get recommendations most accurate for their preferences. Other future work includes addressing the computational expensiveness when dealing large sparse matrices and accessing over 900 thousands posts in a remote database dynamically without compromising user experience. This app as a demonstration uses approx. 175,000 posts a subset of the total raw posts which is totaled over 900 thousand. Another incremental improvement would be to development an iPhone app prototype to complement and synch with this Android app as they can share the same raw Tudiabetes data hosted on the cloud infrastructure.

**Conclusion**

This mobile app prototype demonstrates the possibility of incorporating temporal based recommendations into a social media mobile app. Such functionality can be of great value for social media platforms seeking to increase their user’s participation and collaboration. Moreover, this project demonstrates yet another example of how machine learning can be integrated with cutting-edge mobile technology.