A Seminar Report on

Friendbook: A Semantic-Based Friend Recommendation System for Social Networks

Undergone at

Department of Computer Science & Engineering
National Institute of Technology, Karnataka

Under the guidance of

Ms. Mariet Furtado
Mrs. UmapriyaDharmaraj

Submitted by

Prajwala TM
Reg No: 14CO133
VI SemB.Tech (CSE)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER ENGINEERING

Department of Computer Science & Engineering Technology
National Institute of Technology Karnataka, Surathkal.

March 2017

ABSTRACT

The paper titled “Friendbook: A Semantic Based Friend Recommendation System for Social Networks” by ZhiboWang ,Jilong Liao, Qing CaoHairong Qi, and Zhi Wangpresents Friendbook, a novel-based method for recommendation of friends in social networks, which is done based on lifestyles rather than the traditional method of using social graphs. User-centric data is captured from the mobile phones of the users using sensors, and lifestyles are extracted from it. The similarity between different users is measured based on the similarity between their lifestyles and friends are recommended to the users if they have high similarity. The idea originates from text mining methodologies, which model users’ daily lives as text documents and the lifestyles are extracted from the life documents using the Latent-Dirichlet-Allocation algorithm. The similarity is measured between the lifestyles thus extracted, and a friend matching graph is obtained based on the similarity indices. This is further used for impact ranking, and recommendation scores are obtained for each user. Based on the descending order of the recommendation scores thus obtained, friends are recommended to the users. The impact ranking algorithm is inspired by the PageRank algorithm used by Google.

INDEX

1. Technologies Used ……...1
2. Work Done ……...2
3. Code …… 6
4. Results ……..13
5. Future Scope……14
6. Conclusion ……..15
7. References ……..…..15

TECHNOLOGIES USED

R

R is an open source programming language and software environment for statistical computing and graphics. R and its libraries implement a wide variety of statistical techniques including linear and non-linear modeling, classification, clustering and others. Also, R is highly extensible through the use of user-submitted packages for specific functions.R was used for the implementation of the proposed prediction system.

RStudio

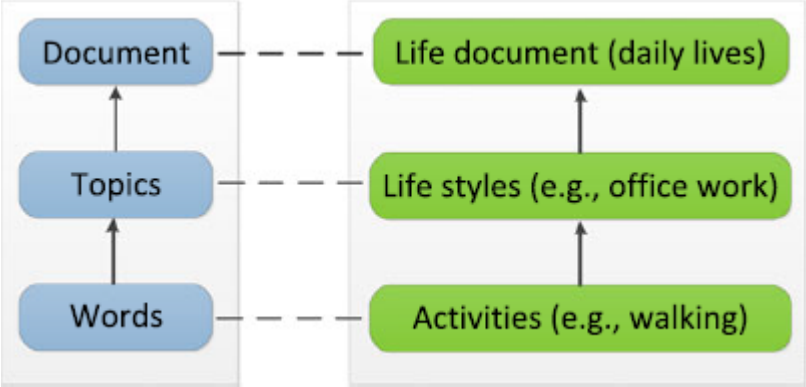
RStudio is a free and open-source integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution and tools for plotting, history, debugging and workspace management. The integration of the above-mentioned features makes it a powerful and productive user interface for R.

WORK DONE

Recent advances in smart phone technologies has inspired the authors of the paper to make use of the inbuilt sensors to extract content-aware information and sensing the daily routines of the users. The raw data however, poses certain challenges to the extraction of the lifestyles in a form useful for text-mining.

Lifestyle Extraction:

The proposed system is inspired by the topic-modelling text mining techniques. The life-documents (daily lives) of the users are modelled as text documents, and these as a mixture of lifestyles modelled as topics, which are further composed of activities modelled as words. This analogy can be represented as follows:



An activity-classifier, on the client side is trained to recognise the activities and report the generated life documents to the servers (K-means can be chosen) . On the server side, the well-known Latent Dirichlet Allocation Algorithm is used for the extraction of the topic-document matrix from the activity-document matrix that is available, using the following equation.

$$p(\mathbf{w} \mid \mathbf{d}) = p(\mathbf{w} \mid \mathbf{z}) p(\mathbf{z} \mid \mathbf{d}).$$

where p(w|d) is the activity-document matrix, p(w|z) is the activity-topic matrix and p(z|d) is the topic-document matrix.

Similarity Metric:

A similarity metric is used to measure the similarity between lifestyles of two users. Given the lifestyle vectors of two users where lifestyle of the jth user is of the form,

$$\mathbf{L}_j = [p(z_1 \mid d_j), p(z_2 \mid d_j), \ldots, p(z_Z \mid d_j)]$$

the similarity is calculated as:

$$S(i,j) = S_c(i,j) \cdot S_d(i,j)$$

Where

$$S_c(i,j) = \cos(\mathbf{L}_i, \mathbf{L}_j)$$

This is done using the cosine similarity metric between the ith and jth lifestyles. The other similarity index is the Dominant lifestyle index. Here, the lifestyle of each user is sorted in descending order and the dominant values are extracted and added to the D_i vector. Finally,

$$S_d(i,j) = \frac{2\,|D_i \cap D_j|}{|D_i| + |D_j|}.$$

Is calculated and used in the similarity equation.

Friend-matching graph construction:

A friend-matching graph is then constructed where each user is a node, and there is an edge between two users if the $S_{i,j}$ is greater than a S_{th}, else the weight is zero. The friend matching graph looks as follows:

$$\mathbf{N} = (N_{ij})_{n\times n} = \begin{bmatrix} 0 & \omega(1,2) & \cdots & \omega(1,n) \\ \omega(2,1) & 0 & \cdots & \omega(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ \omega(n,1) & \omega(n,2) & \cdots & 0 \end{bmatrix}$$

Where the weight of the edge

$$\omega(i,j) = S(i,j)$$

Impact Ranking:

The impact rank of a user, representing the user's capability to establish friends in the network is inspired by the PageRank algorithm, used for ranking webpages. It is constructed on the same lines and a ranking vector is generated which is further used for generation of the recommendation scores.

$$r(i) = \frac{\sum_{j \in \mathcal{N}(i)} \omega(i,j) \cdot r(j)}{\sum_{j \in \mathcal{N}(i)} \omega(i,j)}$$

Where r(i) represents the rank of the ith user computed iteratively. Further changes are made to the above equation for the accuracy improvement purposes, which is explained in detail in the paper.

Query and Friend Recommendation:

The friends are recommended to the user based on the weights given to similarity metric S(i,j) and impact ranks r(i), as follows:

$$R_i(j) = \beta S(i,j) + (1-\beta) r_j \kappa$$

K=n/10 , calculated experimentally for computation purposes. $R_i(j)$ is the recommendation score of jth user to the ith user. The top p scores are used to recommend p friends to the user i where, p is input by the user.

Results:

The results are evaluated using recommendation precision and recommendation recall.

$$R_p = \frac{\sum_i |F_i \cap G_i|/|F_i|}{1{,}000} \quad R_r = \frac{\sum_i |F_i \cap G_i|/|G_i|}{1{,}000} = \frac{\sum_i |F_i \cap G_i|/100}{1{,}000}$$

Due to the unavailability of real-time data, simulation of the results has been carried out as per the guidelines in the paper.

For implementing the proposed prediction system, the following steps have been performed in order:

1. Installation of the latest version of R and RStudio in 64-bit Windows machine.
2. The lifestyle vectors are generated randomly and independently for 1000 users, and normalised individually.

```
for (i in 1:1000) {
lifestyle[i,]<- runif(10)
}
```

3. The cosine similarity and dominant lifestyle similarities are calculated for the lifestyle vectors generated, and finally the S vector is obtained by:

```
S[i,j]=S.c[i,j]*S.d[i,j]
```

4. The top 100 friends are stored separately as true friends, which is used during evaluation.
5. A S_{th} is set and the friend-matching graph is obtained.
6. The ranking vector is obtained for each user using the page rank algorithm in R.

```
r=page.rank (g, vids = V(g), directed = FALSE, damping = 0.85, weights = NULL, options = igraph.arpack.default)$vector
```

7. Recommendation precision and Recommendation recall are calculated for each user.
8. A plot for Recommendation Precision and Recommendation Recall is plotted for a specific beta=0.85 value.

CODE

```
setwd("E:/Computer Engg/Seminar/")
library(igraph)
library(ggplot2)
library("reshape2")
lifestyle<-matrix(NA, nrow=1000, ncol=10)
for (i in 1:1000) {
lifestyle[i,]<- runif(10)
}
head(lifestyle)
for(i in 1:1000) {
sum=0
for(j in 1:10)
  {
sum=sum+lifestyle[i,j]
  }
for(j in 1:10)
  {
lifestyle[i,j]=lifestyle[i,j]/sum
  }
}
s=0
for(j in 1:10)
 s=s+lifestyle[1,j]
S.c<-matrix(NA,nrow=1000,ncol=1000)
for(i in 1:1000)
{
for(j in 1:1000)

S.c[i,j]=crossprod(lifestyle[i,],lifestyle[j,])/sqrt(cros
sprod(lifestyle[i,])*crossprod(lifestyle[j,]))
}
lifestyle.sorted<-matrix(NA,nrow=1000,ncol=10)
```

```
for(i in 1:1000)
lifestyle.sorted[i,]=lifestyle[i,order(-lifestyle[i,])]
dominant.lifestyle<-matrix(NA,nrow=1000,ncol=10)
lambda=0.6;
for(i in 1:1000)
{
sum=0;
for(j in 1:10)
  {
sum=sum+lifestyle.sorted[i,j];
if(sum<lambda)
dominant.lifestyle[i,j]=lifestyle.sorted[i,j];
  }
}
S.d<-matrix(NA,nrow=1000,ncol=1000);
for(i in 1:1000)
{
for(j in 1:1000)
  {

a=length(dominant.lifestyle[!is.na(dominant.lifestyle[i,]
)])

b=length(dominant.lifestyle[!is.na(dominant.lifestyle[j,]
)])

c=length(intersect(dominant.lifestyle[i,],dominant.lifest
yle[j,]));
S.d[i,j]=2*c/(a+b);
  }
}
S<-matrix(NA,nrow=1000,ncol=1000);
for(i in 1:1000)
{
for(j in 1:1000)
S[i,j]=S.c[i,j]*S.d[i,j];
```

```
}
true.friends1<-matrix(NA,nrow=1000,ncol=1000);
for(i in 1:1000)
true.friends1[i,]=order(-S[i,])

true.friends2<-matrix(NA,nrow=1000,ncol=100);
for(i in 1:1000)
{
for(j in 1:100)
true.friends2[i,j]=true.friends1[i,j]
}
sim.thresh=0.0002;
fmg<-matrix(NA,nrow=1000,ncol=1000);
for (i in 1:1000)
{
for(j in 1:1000)
  {
if(i!=j & S[i,j]>sim.thresh)
fmg[i,j]=S[i,j]
else
fmg[i,j]=0;
  }
}
g       <-       graph.adjacency(fmg,       weighted=TRUE,
mode="undirected")
r=page.rank (g, vids = V(g), directed = FALSE, damping =
0.85,
weights = NULL, options = igraph.arpack.default)$vector
k=100
beta=0.6
A = beta*S
A=A+(1-beta)*r*k
r.precision=0
r.recall=0
```

```
rec.prec<-matrix(0,nrow=1,ncol=10);
rec.recall<-matrix(0,nrow=1,ncol=10);
recommended.friends1<-matrix(NA,nrow=1000,ncol=1000);
for(i in 1:1000)
recommended.friends1[i,]=order(-A[i,])
l=1
for(p in c(100,200,300,400,500,600,700,800,900,1000))
{
  recommended.friends2<-matrix(NA,nrow=1000,ncol=p);
for(i in 1:1000)
{
for(j in 1:p)
recommended.friends2[i,j]=recommended.friends1[i,j]
}
for(i in 1:1000)
{
    a=length(true.friends2[i,])
    b=length(recommended.friends2[i,])

c=length(intersect(true.friends2[i,],recommended.friends2
[i,]));
rec.prec[1,l]=rec.prec[1,l]+c/b;
rec.recall[1,l]=rec.recall[1,l]+c/a;
    #r.precision=r.precision+c/b;
    #r.recall=r.recall+c/a;


}
rec.prec[1,l]=rec.prec[1,l]/10
rec.recall[1,l]=rec.recall[1,l]/10
l=l+1
}
r.precision=r.precision/1000
r.recall=r.recall/1000
number=c(100,200,300,400,500,600,700,800,900,1000)
plot(number,t(rec.prec),type = "o", col = "red", xlab =
```

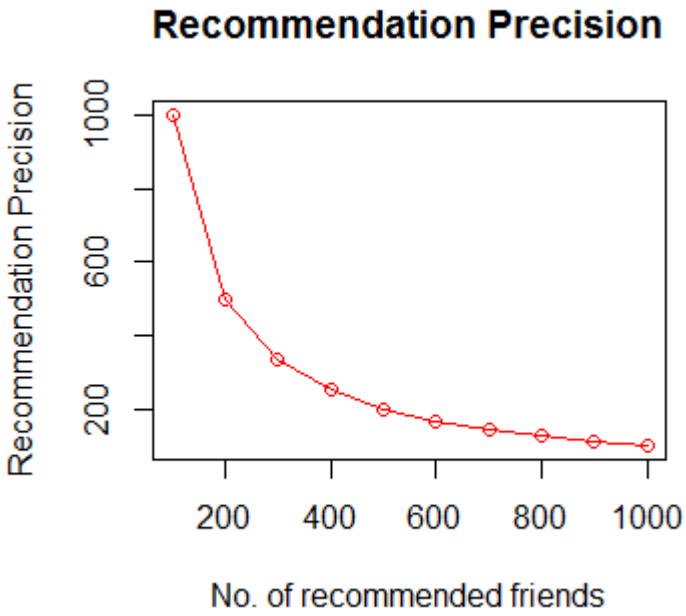
```
"No. of recommended friends", ylab ="Recommendation Precision" ,
main = "Recommendation Precision")
plot(number,t(rec.recall),type = "o", col = "red", xlab = "No. of recommended friends", ylab ="Recommendation Recall" ,
main = "Recommendation Recall")
```

RESULTS

The recommendation precision and recommendation values are calculated and compared with those obtained in the paper for a particular beta value = 0.85 and graphs are plotted.

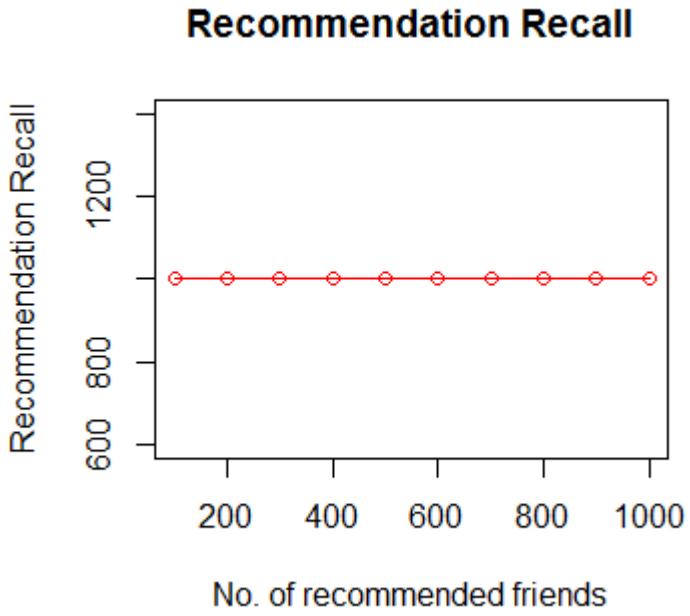
Recommendation Precision:

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
100	50	33.33333	25	20	16.66667	14.28571	12.5	11.11111	10



Recommendation Recall:

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
100	100	100	100	100	100	100	100	100	100



FUTURE SCOPE

First, the system can be evaluated on large-scale field experiments. Second, implementation of life style extraction using LDA and the iterativematrix-vector multiplication method in user impact ranking can be done incrementally, so that Friendbook would be scalable to large-scale systems. Third, the similarity threshold used for the friend-matching graph is fixed in our current prototype of Friendbook. It would be interesting to explore the adaption of the threshold for each edge and see whether it can better represent the similarity relationship on the friendmatching graph. At last, we plan to incorporate more sensors on the mobile phones into the system and also utilize the information from wearable equipments (e.g., Fitbit, iwatch, Google glass, Nike+, and Galaxy Gear) to discover more interesting and meaningful life styles, while using real-time data.

CONCLUSION

The results obtained using the simulation matched with those obtained in the paper. The results show that friends are recommended with a good amount of accuracy, in comparison to those recommended using social graphs.

REFERENCES

1) http://igraph.org/r/doc/page_rank.html
2) http://www.statmethods.net/advgraphs/ggplot2.html
3) http://www.pagerank.dk/Pagerank-formula/Damping-factor.htm
4) https://stat.ethz.ch/R-manual/R-devel/library/base/html/data.frame.html
5) https://stats.idre.ucla.edu/r/library/r-library-matrices-and-matrix-computations-in-r/
6) https://www.tutorialspoint.com/r/