Friendbook: A Semantic-based Friend Matching System for Social Networks

Presented by Qing Cao
In collaboration with Zhibo Wang, Jilong Liao, Hairong Qi, and Edward Taylor

Supported by a NSF funding on “distributed smart camera networks”
Outline

• Introduction

• Friendbook Framework
  – Sensing
  – Activity inference
  – Interest extraction
  – S-friend matching

• Prototype Implementation

• Performance Evaluation

• Conclusions and Future work
Motivation

• Traditional way of making friends (G-friend)
  – Geographical location based: Neighbors, colleagues
  – Pros: be familiar with each other
  – Cons: # of friends is limited

• Emerging social networks
  – Facebook, Twitter, Google+, Renren, etc
  – Pros: unlimited # of “friends”
  – Cons: “Friends” are not the expected friends

➢ What’s the fact?
  ➢ People tends to make friends with people having similar interests
  ➢ E.g., like travelling, riding, shopping
Motivation

• Problems:
  – How to identify friend candidates based on interests rather than pre-existing relationships?
  – How to automatically get one’s interests without one’s specification?
  – How to help people find friends at any time and any place?
  – How to protect the personal privacy during friend making?
  – How to measure the similarity of interests between different users?

• Objective:
  – We would like to design and implement a semantic-based friend matching system that allows users with similar interests to be quickly identified and recommended while preserving users’ privacy
Mobile Phone

• Our solution is motivated by the advances in smart phones

• Why do we choose mobile phone?
  – Equipped with multiple embedded sensors
    • GPS, Accelerometer, Camera, Gyroscope, Light, etc.
  – Popular, portable and widely used
  – An ideal mobile sensing platform for capturing daily route

Nexus S

iPhone 4
Objective:
- We would like to design and implement a semantic-based friend matching system on mobile phones for social networks that allows users with similar interests to be quickly identified and recommended with preserving users' privacy.
Challenges

• How should we design effective algorithms to extract interest information on the client side when the algorithm has to run on battery-powered mobile phones?

• How should the server process the user’s interest query?

• How to protect the privacy of users during friend matching?
Mobile Sensing

• Function: raw data collection

• Multi-modal sensors:
  – GPS, Accelerometer, Gyroscope, Camera, etc.
  – Suppose n sensors are used, $x_i$ is the data recorded by the i-th sensor
  – Data vector: $x = [x_1, x_2, \ldots, x_n]$ represent sensing results

• User has the control of enabling or disenabling sensors
  – Privacy consideration
  – Energy consideration

• In our implementation:
  – Accelerometer and Camera are exploited
  – More sensors can be integrated
Activity Inference

• Function:
  – Identify one’s activity based on the raw data from sensing phase

• What’s activity?
  – One’s status or action at a short time period
  – Motion activity:
    sitting/walking/running/riding/driving

• Two sub-phase
  – Feature extraction
  – Activity recognition
Feature Extraction

• Feature:
  – A mathematical description of raw data
  – Characterize the salient properties of the raw data but with much lower dimension

• Function:
  – Extracts representative features from raw data and provides these features to classifier for further analysis
  – Feature-based activity recognition is more computationally efficient and more robust compared to raw data-based recognition

• In our implementation
  – Feature extraction on Accelerometer
  – Feature extraction on Camera
Feature Extraction on Accelerometer

- Accelerometer is used to identify five modes of motion activities: sitting, walking, running, riding, driving
Feature Extraction on Accelerometer

- Accelerometer is used to identify five modes of motion activities: sitting, walking, running, riding, driving.

Notice that standard deviation provides more distinct description of the five different motion activities and thus would serve as a better feature compared to mean.

Feature vector extracted from accelerometer: \( \mathbf{f}_{\text{act}} = [\sigma_x, \sigma_y, \sigma_z] \).
Feature Extraction on Camera

• Camera:
  – We know we have camera and we use it to take pictures
  – “A picture is worth a thousand words”.
  – Different pictures imply different interests of people.

• Feature points can be extracted to provide a “feature description” of a picture.
Feature Extraction on Camera

• Feature extraction algorithm:
  – Scale Invariant Feature Transform (SIFT) algorithm
  – Speed-Up Roust Features (SURF) algorithm

We adopt SURF algorithm in Friendbook to extract feature vectors as it requires less computation resources than SIFT.

Feature vector extracted from a picture:

\[ \mathbf{f}_{img} = [\alpha_1^1, \cdots, \alpha_1^{64}, \alpha_2^1, \cdots, \alpha_2^{64}, \cdots, \alpha_l^1, \cdots, \alpha_l^{64}] \]
Activity Recognition

• Function:
  – Label each feature vector with a recognized activity

• Supervised classification algorithms are applied here:
  – Decision tree learning
  – Naïve bayes classifier
  – Support vector machine, etc.

• The decision tree classifier is trained off-line based on labeled feature vectors and then used to classify unlabeled feature vectors in a real-time manner.

  Activity sequence vector: \( \mathbf{a} = [a_1, \ldots, a_m] \) \( i=1, \ldots m \) is the time index

Note that we do not further process \( f_{img} \) as it already implicitly reflect one’s interest
Interest Extraction

• What is interest?
  – Compared to activity, interest is a higher level of abstraction for human behavior, as it reflects one’s preference on certain things or passion in doing certain things.

• Activity and interest are closely related
  – Running $\rightarrow$ like running
  – Walking + green way $\rightarrow$ like walking on the green way

• We propose to infer one’s interests by mining of temporal correlation patterns among activities

Note that for friend matching purpose based on interest, our main task is not to identify interests, rather we need to differentiate between different interests. This observation has greatly simplified the development process.
Interest Extraction (cont’d)

• A simplified shape statistics method is used to extract interest:

\[ \mathbf{l}_{\text{act}} = [\mu_{\text{shape}}, \theta_{\text{shape}}, \gamma_{\text{shape}}, \beta_{\text{shape}}] \]

Purposes:

➢ Extracts salient information that characterizes the difference in temporal patterns among activities to uniquely identify different interests
➢ Provides the second layer protection regarding user privacy as the interest vector can not be back mapped to the exact activity sequence
Interests Extraction

• Function:
  – The last phase on mobile phone that extracts interests from patterns of activities

• Interests representation:
  – Shape statistics: mean, standard deviation, skewness, koutorsis
  – Numerical activity privacy protection
  – Preserve sequence information

• Interests will be automatically reported to the server
### One Example

<table>
<thead>
<tr>
<th>Interest</th>
<th>Like walking on the green way</th>
<th>Like running</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor reading</td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
</tr>
<tr>
<td>Feature vectors</td>
<td>${2.8, 3.2, 2.3}, \ldots, {2.2, 3.7, 3.3}$</td>
<td>${7.2, 5.8, 5.4}, \ldots, {7.9, 7.1, 4.2}$</td>
</tr>
<tr>
<td>Activity sequence</td>
<td>${1, 1, 1, \ldots, 1}$</td>
<td>${2, 2, 2, \ldots, 2}$</td>
</tr>
<tr>
<td>Interest vector</td>
<td>$\langle 4.5, 12.3, -0.32, 0.1 \rangle$</td>
<td>$\langle 9, 3.81, -0.22, 0.05 \rangle$</td>
</tr>
</tbody>
</table>
S-friend Matching

• Interest vectors $I_{act}$ and $I_{img}$ are uploaded from clients to the server.

• S-friend matching aims to measure the similarity of interest vectors of different users and reply a ranked friend list based on the similarity score to users.

• Challenges:
  – How to measure the similarity of interest vectors?
  – How to calculate the similarity score of different users?

• Two components:
  – Similarity measurement
  – Friend matching
Similarity Measurement based on $I_{act}$

- Similarity measure of interests:
  \[
  s(I^i_{act}, I^j_{act}) = \frac{1}{1 + d(I^i_{act}, I^j_{act})}
  \]

  Similarity score is between 0 and 1. The more similar the two interest vectors, the close similarity score is to 1.

- Similarity measure of users:
  \[
  sim(user_i, user_j) = \frac{\max_{r,t} \{\sum_{k=1}^{p} s_k(I^r_{act}, I^t_{act})\}}{p}
  \]
Optimal Matching Algorithm

Algorithm 1 Match-Single-User-Pair

**Input:** Any two users’ interest sequence $A(i), i = 1, 2, \cdots, n$ and $B(j), j = 1, 2, \cdots, m$

**Output:** The optimal normalized similarity score:

\[
\frac{dp(n,m)}{pairs(n,m)}
\]

1: $dp(0, \cdots, n) \leftarrow 0$
2: $dp(0, \cdots, m) \leftarrow 0$
3: $pairs(0 \cdots n, 0 \cdots m) \leftarrow 0$
4: for $i \leftarrow 1$ to $n$ do
5: \hspace{1em} for $j \leftarrow 1$ to $m$ do
6: \hspace{2em} $d \leftarrow \text{Euclidean-Distance}(A(i), B(j))$
7: \hspace{2em} $s \leftarrow \frac{1}{1+d}$
8: \hspace{2em} find-max\{ $dp(i-1, j-1) + s$, $dp(i,j-1), dp(i-1,j)$ \}
9: \hspace{2em} if $dp(i-1, j-1) + s$ is the max then
10: \hspace{3em} $dp(i, j) \leftarrow dp(i-1, j-1) + s$
11: \hspace{3em} $pairs(i, j) \leftarrow pairs(i-1, j-1) + 1$
12: else if $dp(i-1, j)$ is the max then
13: \hspace{3em} $dp(i, j) \leftarrow dp(i-1, j)$
14: \hspace{3em} $pairs(i, j) \leftarrow pairs(i-1, j)$
15: else
16: \hspace{3em} $dp(i, j) \leftarrow dp(i, j-1)$
17: \hspace{3em} $pairs(i, j) \leftarrow pairs(i, j-1)$
18: return $\frac{dp(n,m)}{pairs(n,m)}$
Friend Matching

• Top k results are recommended
• Complete user list traversal

Algorithm 2 Top-k-Recommendations

Input: A user’s query: \( Q \)
Output: The top k recommendation results: \( recom(1 \cdots k) \)

1: Query database and construct user interest sequence: \( A(1 \cdots n) \)
2: \( recom \leftarrow \text{new Priority-Queue}() \)
3: \( pool \leftarrow \text{full-user-list}() \)
4: for all user in pool do
5: \( score \leftarrow \text{Match-Single-User-Pair}(A, user) \)
6: insert \( (user.name, score) \) to \( recom \)
7: adjust \( recom \) and pop the tail
8: return \( recom \)
**Friend Matching (cont’d)**

- Merging results:

---

**Algorithm 3 Merge-Rankings**

**Input:** Two friend rankings from match engine: $R_1, R_2$

**Output:** The overlap of two rankings: $\textit{final}$

1: $\textit{final} \leftarrow \emptyset$

2: **for all** result in $R_1$ **do**

3: **if** find result in $R_2$ **then**

4: insert into $\textit{final}$

5: **return** $\textit{final}$

---
System Implementation

• Client implementation:
  – Nexus S: Android 2.3.6 operating system
  – 2969 lines Java code and 389 lines of XML code

Client application architecture
System Implementation

- Server implementation:
  - 619 lines Python code
  - Dedicated Dell workstation with Linux operating system

Server architecture
System Performance

- The performance are evaluated at the following metrics:
  - Activity classification
  - Picture matching
  - Simulation results of match engine
  - In-field matching performance
  - System response time
  - Energy consumption
Activity Classification

- Motion activities: sitting/walking/running/riding/driving

Table 1: Confusion matrix for activity classification

<table>
<thead>
<tr>
<th>Actual Activity</th>
<th>Sit</th>
<th>Walk</th>
<th>Run</th>
<th>Ride</th>
<th>Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sit</td>
<td>93.1%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>6.9%</td>
</tr>
<tr>
<td>Walk</td>
<td>0%</td>
<td>87.8%</td>
<td>7.0%</td>
<td>1.8%</td>
<td>3.4%</td>
</tr>
<tr>
<td>Run</td>
<td>1.1%</td>
<td>6.7%</td>
<td>88.8%</td>
<td>2.2%</td>
<td>1.2%</td>
</tr>
<tr>
<td>Ride</td>
<td>1.8%</td>
<td>13.1%</td>
<td>2.8%</td>
<td>76.7%</td>
<td>5.6%</td>
</tr>
<tr>
<td>Drive</td>
<td>19.7%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>3.9%</td>
<td>72%</td>
</tr>
</tbody>
</table>
Picture Matching

(a) Picture 1  (b) Picture 2  (c) Picture 3  (d) Picture 4

Table 2: Similarity matrix for four pictures

<table>
<thead>
<tr>
<th></th>
<th>Picture 1</th>
<th>Picture 2</th>
<th>Picture 3</th>
<th>Picture 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Picture 1</td>
<td>100%</td>
<td>67.6%</td>
<td>1.7%</td>
<td>1.3%</td>
</tr>
<tr>
<td>Picture 2</td>
<td>67.6%</td>
<td>100%</td>
<td>1.1%</td>
<td>13.5%</td>
</tr>
<tr>
<td>Picture 3</td>
<td>1.7%</td>
<td>1.1%</td>
<td>100%</td>
<td>33.6%</td>
</tr>
<tr>
<td>Picture 4</td>
<td>1.3%</td>
<td>13.5%</td>
<td>33.6%</td>
<td>100%</td>
</tr>
</tbody>
</table>
In-field Matching

- Eight volunteers with different interests
- Strong connection: two people share more than half of their overall interest
- Weak connection: larger than a threshold, but less than half

Ground truth

Online test
System Response Time

Table 3: Response time for each operation

<table>
<thead>
<tr>
<th>Operation</th>
<th>Client side (ms)</th>
<th>CPU time in Server (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>Turn sensor on</td>
<td>43</td>
<td>–</td>
</tr>
<tr>
<td>Turn sensor off</td>
<td>17</td>
<td>–</td>
</tr>
<tr>
<td>Activity classification</td>
<td>24</td>
<td>–</td>
</tr>
<tr>
<td>Interest extraction</td>
<td>1</td>
<td>–</td>
</tr>
<tr>
<td>Interest Upload</td>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>Image processing</td>
<td>7092</td>
<td>–</td>
</tr>
<tr>
<td>Image feature upload</td>
<td>193</td>
<td>21</td>
</tr>
<tr>
<td>Friend Query</td>
<td>6918</td>
<td>902</td>
</tr>
</tbody>
</table>
# Energy Consumption

**Table 4: Standard usage pattern**

<table>
<thead>
<tr>
<th>Duration (minutes)</th>
<th>Normal Operation</th>
<th>ProjectX deployed</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>Gmail</td>
<td>Gmail</td>
</tr>
<tr>
<td>10</td>
<td>–</td>
<td>ProjectX</td>
</tr>
<tr>
<td>20</td>
<td>Facebook</td>
<td>Facebook</td>
</tr>
<tr>
<td>10</td>
<td>–</td>
<td>ProjectX</td>
</tr>
<tr>
<td>20</td>
<td>Google Maps</td>
<td>Google Maps</td>
</tr>
<tr>
<td>10</td>
<td>–</td>
<td>ProjectX</td>
</tr>
<tr>
<td>20</td>
<td>Youtube</td>
<td>Youtube</td>
</tr>
<tr>
<td>10</td>
<td>–</td>
<td>ProjectX</td>
</tr>
</tbody>
</table>
Energy Consumption

(a) Battery Level

(b) Energy Consumption
Conclusions

• To the best of our knowledge, Friendbook is the first to present a novel friend matching system based on mobile phones for social networks.

• The proposed friend matching method is semantic-based instead of conventional keyword-based or relation-based as most existing networks adopt.

• Instead of performing only activity recognition, we further exploit temporal correlations among activities to infer interests of human being.

• We propose a two-tiered privacy preserving mechanism at the user side to protect user privacy.

• We propose an effective similarity metric for measuring the similarity of interests between different users, as well as a dynamic programming search algorithm to find the similarity of any pair of users.

• We implement a prototype system using Nexus S mobile phones and evaluation demonstrates the effectiveness of proposed approaches.