A Social Ringer Manager Application

CSC 555 Fall 2015

The purpose of a ringer application is to notify a user of an incoming call by ringing. In a typical setting, user presets a ringer mode, and the phone rings according to the preset ringer mode. However, you may have observed ringer mode depends upon multiple attributes, such as the place where you are, noise level at the place, who are your neighbors, who is the caller, caller's urgency, etc. Further, your phone ringing loud or not ringing at all during certain scenario might lead to negative or positive feedbacks by your neighbors and the person trying to call you. In this assignment, your task is to develop a social ringer manager application that adapts to requirements crowdsourced from multiple information sources around you. The following scenarios exemplify a few opportunities for adaptation.

- Alice is in the library and she receives a call from her friend, Bob. Alice's phone ringer is set to 'loud' and she ends up disturbing her neighbors. This might lead to Alice's neighbors giving her a negative feedback.
- Alice is in the gym and she receives an urgent call from her father, Dave. Her phone ringer is set to 'loud'. Since the gym is noisy, phone ring does not bother Alice's neighbors. They send her a neutral feedback.
- Alice is at a party with her friends and her friend, Bob, receives a call from someone at work which isn't urgent. Bob ignores the call. Alice and her other friends are happy with Bob not answering the call, and hence send Bob a positive feedback.

Contextual Information Sources

The following are the contextual information sources you have to employ in the social ringer manager application.

User Context

- User's location
- Surrounding noise level
- People surrounding user
- What is their relation to user?
  - Type (family, friend, colleague, or stranger)

  - Strength of relation (high, medium, or Low) [This can be defined by you]

- What is the ringer mode they have?
- What ringer mode do they expect you to have?

Caller Context

- What is their relation to user?
  - Type (family, friend, colleague, or stranger)
  - Strength of relation (high, medium, or Low) [This can be defined by you]
- Urgency of the call (yes or no)

Based on the user and caller contextual information sources, you have to come up with a *utility* function that determines whether user's phone should ring for a particular call or not.

Adaptation

Based on decision taken by you (your utility function) for a particular call, your neighbors might provide you with positive, negative, or a neutral feedback. Your utility function should also consider past feedbacks.

Implementation

You will implement your application using Twitter API. Your application should be able to send and receive the messages listed below. Please ensure you follow the format correctly otherwise you will not receive responses to your tweets. For testing purposes we have a Twitter Bot that is monitoring #P2CSC555F15 that replies to tweets sent out with the formats given below. Later, this bot will be replaced by actual applications.

Bot List. Anakin (Family), Chewbacca (Friend), Han (Friend), Jango (Stranger), JarJar (Friend), Leia (Family), Luke (Family), Mace (Colleague), ObiWan (Colleague), Padme (Family), Yoda (Colleague)

Location List. #hunt, #eb2, #carmichael, #oval, #party

Message Unique Identifier. All tweets should end with "#id_unityID_num #P2CSC555F15" where unityID is your unityID, and num is the check-in message number (1 for the first check-in, 2 for the second, etc.).

Check-in. Once you enter to a place, you should tweet your check-in.

Syntax. "I checked in at #<location>#id_unityID_num #P2CSC555F15."

Listing 1: Response to check-in from neighbors

```
@sender
Name: <Bot name>
MY_MODE: Silent/Loud
EXPECTED_MODE: Silent/Loud
#id_unityID_num #P2CSC555F15
```

Listing 2: Response to check-in from place where you checked-in

```
@sender
LOCATION: <location>
NOISE: <noise level> (1-5, where 1 indicates silence and
                      5 indicates a noisy environment)
#id_unityID_num #P2CSC555F15
```

Your program will also have to continuously listen for check-in messages from other users and respond appropriately to them with the message format for the response mentioned above.

Make Calls. Once you check-in you can request an incoming call. This is to test the case where you receive a phone call.

Syntax. "CALL #id_unityID_num #P2CSC555F15"

Listing 3: Response to call

```
@sender
Call from: <Bot name>
URGENCY: 0/1
#id_unityID_num #P2CSC555F15
```

Call Response. For each call you receive, you will have to reply with a message indicating what action was taken. There are two choices for this, (1) Yes– indicating that incoming was answered, or (2) No– indicating the call was silenced.

Syntax. "ACTION: Yes/No' #id_unityID_num #P2CSC555F15"

In response to your action on call, your neighbors will reply back indicating whether they were happy or not with your action.

Listing 4: Neighbor feedback

```
@sender
Name: <Neighbor Name>
RESPONSE: Positive/Negative/Neutral
\#id\_unityID\_num \#P2CSC555F15
```

Deliverables

Phase A

- Pre-project survey *
- Utility function
- Intermediate project source
- Read-me on how to execute the application
- Intermediate report
- Time and effort survey for each work-session *

Phase B

- Final project source
- Read-me on how to run the application
- Final report
- Time and effort survey for each work-session *
- Post-project survey *

* If you consented to participate in the study.