Exploiting Latent Information to Predict Diffusion of Novel Topics on Social Networks

Introduction

This is the readme for the resource of the "Exploiting Latent Information to Predict Diffusion of Novel Topics on Social Networks" research, including **dataset** and **source code**.

Software Platform

We use Java as our development platform. Please install **Java 1.6.0** or above in order to execute or modify our source code.

Hardware Platform

We suggest using AMD Opteron 2350 2.0GHz Quad-core CPU or above, 32GB RAM or above, and 500GB hard disk space to run the program.

Acknowledgement

We used these libraries in our program (in alphabetical order):

Apache Common Math 1.2 or above	http://commons.apache.org/math
Apache Collections-Generic 4.0.1 or above	http://larvalabs.com/collections/
AUC Calculator 0.2 or above	http://mark.goadrich.com/programs/AUC/
BLAS 0.8 or above	http://www.netlib.org/blas/
JUNG 2.2.0 or above	http://jung.sourceforge.net/
LIBLINEAR 1.5.1 or above	http://www.csie.ntu.edu.tw/~cjlin/liblinear/

Getting Started

1. Please extract “Software.zip”. Several important directories are listed below:

src	Source code of our program
jar	JAR file of our program (for quick start)
lib	Directory for required libraries
data	Data files (the content of “Dataset.zip should be put under this directory)

2. Please extract "Dataset.zip" and copy the files and directories to **data/plurk_iii**.

3. Among the source codes in the ***src*** directory or JAR file in the ***jar*** directory, five Java classes are designed to be run directly, and the results are displayed in system console:

PrepareData.java	Main class for preparing data
BaselineDF.java	Main class for baseline diffusion number for existing topics
BaselineIC.java	Main class for baseline Independent Cascade (IC) model
BaselineHD.java	Main class for baseline Heat Diffusion (HD) model
LearningBased.java	Main class for our proposed supervised learning framework

4. First, please execute ***PrepareData*** class to generate intermediate data. There is no parameter for this class. A command line example is shown below. It should be noted that the separating symbol of the *-cp* argument is ***colon*** for some operating systems (e.g., UNIX), and ***semicolon*** for other operating systems (e.g., Windows). We use colon in our examples. Also, the versions of downloaded libraries might differ from those used here.

```
java -Xms512M -Xmx32G
-cp .:./jar/dp.jar:./lib/commons-math-1.2.jar:./lib/collections-generic-4.01.jar:./lib/auc.jar
:./lib/blas-0.8.jar:./lib/jung2-2_0_1.jar:./lib/liblinear-1.51.jar:./lib/utility.jar PrepareData
```

5. Second, please execute ***BaselineDF*** class to predict diffusions using the diffusion number for existing topics. There is no parameter for this class. An example is as below.

```
java -Xms512M -Xmx32G
-cp .:./jar/dp.jar:./lib/commons-math-1.2.jar:./lib/collections-generic-4.01.jar:./lib/auc.jar
:./lib/blas-0.8.jar:./lib/jung2-2_0_1.jar:./lib/liblinear-1.51.jar:./lib/utility.jar BaselineDF
```

6. Next, please execute ***BaselineIC*** class to predict diffusions using the IC model. There is no parameter for this class. An example is as below.

```
java -Xms512M -Xmx32G
-cp .:./jar/dp.jar:./lib/commons-math-1.2.jar:./lib/collections-generic-4.01.jar:./lib/auc.jar
:./lib/blas-0.8.jar:./lib/jung2-2_0_1.jar:./lib/liblinear-1.51.jar:./lib/utility.jar BaselineIC
```

7. Then, please execute ***BaselineHD*** class to predict diffusions using the HD model. The only parameter is the diffusion time. In our experiment, we found the best parameter value = 100. An example is as below.

```
java -Xms512M -Xmx32G
-cp .:./jar/dp.jar:./lib/commons-math-1.2.jar:./lib/collections-generic-4.01.jar:./lib/auc.jar
:./lib/blas-0.8.jar:./lib/jung2-2_0_1.jar:./lib/liblinear-1.51.jar:./lib/utility.jar BaselineHD
100
```

8. Finally, please execute ***LearningBased*** class to predict diffusions using our proposed framework (the core class of our method is **classifier.ClassifierCV**; most instance / feature related information can be found in this class). The only parameter used in our method is the feature combination, which is encoded using a **7-digit number**, composed by **0 and 1**. The features are as below:

Feature **1**:	*Topic Signature (TG)*
Feature **2**:	*Topic Similarity (TS)*
Feature **3**:	*User Signature (UG)*
Feature **4**:	*User Preferences to Latent Categories (UPLC)*
Feature **5**:	*In-degree (ID)*
Feature **6**:	*Out-degree (OD)*
Feature **7**:	*Number of Distinct Topics (NDT)*

For example, the parameter "0001101" indicates to perform learning-based prediction using the feature set UPLC + ID + NDT. An example is as below.

```
java -Xms512M -Xmx32G
-cp .:./jar/dp.jar:./lib/commons-math-1.2.jar:./lib/collections-generic-4.01.jar:./lib/auc.jar
:./lib/blas-0.8.jar:./lib/jung2-2_0_1.jar:./lib/liblinear-1.51.jar:./lib/utility.jar
LearningBased 0001101
```

Software Note

1. Please include all prerequisite libraries in the Java class path (e.g., use *-cp* command line argument to run Java).

2. Please set memory-related arguments (ex. use *-Xms512M -Xmx32G* to run Java). We suggest using 32GB RAM and 500GB hard disk space to run the program.

Dataset

1. The data files are located in the **data/plurk_iii** folder:

data/plurk_iii	Raw data
data/plurk_iii/cvX	Preprocessed data for 4-fold cross validation (***X*** = folder)
data/plurk_iii/negative	Sampled negative instance links

2. **Raw data.** There are five types of files in the folder (the format are in **green brackets**):

   - ***topic_list.txt*** is the ist of 100 topics: **[topic_id] [topic name and/or URL]**
   - ***user_plurk.txt*** is the users in the *original* social network: **[user_id] [nickname]**
   - ***relation_plurk.txt*** is the relations in the *original* social network: **[user_id] [friend_id]**
   - ***message_conceptT_plurk.txt*** is the messages for a topic (***T*** = topic_id): **[message_id] [user_id] [content] [time] [number_of_responses] [number_of_likes] [topic _id]**
   - ***response_conceptT_plurk.txt*** is the responses for a topic (***T*** = topic_id): **[response_id] [message_id] [user_id] [content] [time] [topic_id]**

3. **Preprocessed data.** There are two sub-folders (**train** and **test**). Each of them contains two types of files (using train as example):

   - ***sorted_T_train.txt*** is the preprocessed responses for a topic (***T*** = topic_id): **[source_user_id] [destination_user_id] [time] [content]**
   - ***dn_T_train.txt*** is the summarized information of responses on links for a topic (***T*** = topic_id): **[source_user_id] [destination_user_id] [diffusion_number] [total_diffusion_content_length] [diffusion_number_with_URL] [diffusion_number_with_IMG]**

4. **Negative instances.** There are fives sub-folders (**cv1**, **cv2**, **cn3** and **cv4** is for testing, and **train_unweighted** is for training). All files share the same format:

   - ***dn_T_test.txt*** is the sampled negative links for a topic (***T*** = topic_id, note that the **positive_or_negative** field is always **0**): **[source_user_id] [destination_user_id] [positive_or_negative]**

Please let us know if you have any question or suggestion. We appreciate your time for using our dataset and source code.