Incorporating User Grouping into Retweeting Behavior Modeling*

Author 1† Author 2‡

Abstract

Social media applications are emerging, with rapidly growing users and large numbers of retweeted blogs every minute. The variety among users makes it difficult to model their retweeting activities. Obviously, it is not suitable to cover the overall users by a single model. Meanwhile, building one model per user is not practical. To this end, this paper presents a novel solution, of which the principle is to model the retweeting behavior over user groups. Our approach, GruBa, consists of three key components for extracting user based features, clustering users into groups, and modeling upon each group. Particularly, we look into the user interest from different perspectives including long-term/recent interests and explicit/implicit interests, which results deep analyses towards the retweeting behavior and proper models in the end. We have evaluated the performance of GruBa by datasets of real-world social networking applications and a number of query workloads, showcasing its benefits.

1 Introduction

Social media is overwhelming nowadays, with massive users on Facebook [reference], Twitter [reference] and Weibo [reference] while the number of users keeps increasing. These users behave variously, knowledge of which is significant in recommendation system, activity prediction and Big Thing analysis. Hence emerges the demand of developing systems and algorithms that could properly model user behaviors, which has attracted the attention from both academia and industry.

Central to user behavior modeling, is the need to choose the unit of model (i.e., how many users share one model), as well as the variety of features to be selected for differentiating these units. Already, there exist work of building a single model for all the users [reference]. Apparently, such model bears the limitation of being coarse. On the other hand, modeling each user is not practical, due to the tremendous number of users.

The key driver of our work is the realization that in social media applications, users could fall into groups and each group shares representative behaviors. Particularly, we study the retweeting behavior of users and our work can be generalized to other behaviors of like and comment as well. As one example, consider the film *Brave Heart*, fans of which are probably addicted to highland, bagpipe and war films, and thus likely to retweet blogs of these topics.

The contributions of this work include:

- We present a system named GruBa with the novel perspective to model user behaviors over groups instead of the mono model in literature.
- We leverage user interests to facilitate the modeling of retweeting behavior and look into interests with various dimensions, including long-term/recent interests and explicit/implicit interests.
- We evaluate the performance of GruBa using real-world datasets, showcasing its benefits against competitive state of the art approaches.

The rest of this paper is organized as follows. Section 2 first gives the problem formulation and subsequently overviews GruBa's components, principle of which are detailed in Sections 3, 4 and 5 separately. Section 6 provides the performance evaluation. Related work is presented in Section 7. Finally, Section 8 concludes the work.

2 GruBa Overview

2.1 Problem Formulation We consider people's retweeting behavior in social media. For simplicity, with a given user, we assume that blogs created or retweeted by his/her followees cover the overall candidates, from which the said user may retweet. All our results could straightforwardly generalize to alternative candidate scopes.

DEFINITION 1. *A blog $B = (O, T, M, R)$ consists of the owner O (a.k.a. user in this paper) to whom B belongs (either created or retweeted), the timestamp T showing when B is generated, the blog message M and a bit R denoting B is retweeted (1) or originally created (0) by the owner O.*

*Supported by NSFC (U1636210), 973 program (2014CB340300), NSFC (61322207&61421003), Special Funds of Beijing Municipal Science & Technology Commission, Beijing Advanced Innovation Center for Big Data and Brain Computing, and MSRA Collaborative Research Program.

†affiliation

‡affiliation

DEFINITION 2. *A user* $U = (B_s, R_s, E_s)$ *consists of three sets regarding the user's blogs* B_s*, followers* R_s *and followees* E_s *separately. Each follower/followee per se refers to a user.*

The mapping between blog B and user U is a bilateral operation, i.e., $U = O(B)$ and $B \in B_s(U)$, through ID(s) of user and blog respectively.

Informally, providing a set of users $\{U\}$ and the associated blogs $\{B\}$, as well as a blog query b and a follower of $O(b)$ written as f, i.e., $f \in R_s(O(b))$, GruBa shall build a retweeting model for $(\{U\}, \{B\})$, upon which Y/N is returned regarding whether f shall retweet b.

2.2 GruBa Framework GruBa is designed from the ground up as a system for modeling users' retweeting behavior in social media. Figure 1 shows the architectural components of GruBa, comprising three subsystems: Data Storage, Processing Runtime and Profile Demonstrator.

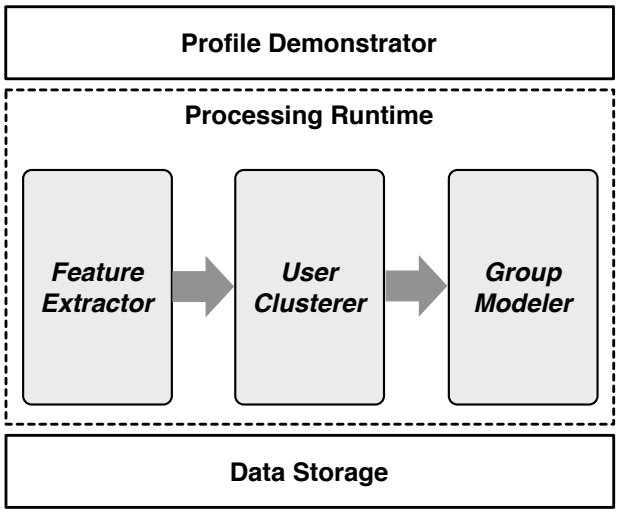

Figure 1: GruBa Architecture

Data Storage. The underlying Data Storage subsystem stores data to be processed by GruBa, i.e., data of blogs and users, as shown in *Definition 1* and *2*.

Processing Runtime. In the heart of GruBa lies the Processing Runtime subsystem, which consists of three major components as follows.

1. Feature Extractor: By coalescing the blog data, each user is depicted by a bunch of features, which are grouped into three categories. They are features of *Info* (e.g., the number of followers and followees), *Behavior* (e.g., the frequency and the popular slots of retweeting) and *Interest* (e.g., the long-term/recent interests, as well as the explicit/implicit interests). These features are extracted from the stored data by Feature Extractor and serve as the input of User Clusterer.
2. User Clusterer: Providing the user-based features, User Clusterer takes charge of the clustering task such that each user falls into a proper group.
3. Group Modeler: For each group obtained by User Clusterer, Group Modeler employs both positive and negative samples to train a model, over which the testing of users' retweeting behavior is performed.

Profile Demonstrator. At the top layer of GruBa, it is the Profile Demonstrator subsystem for visualization. For the time being, Profile Demonstrator presents [To Be Completed].

3 Feature Extractor

With the underlying data in Data Storage subsystem, Feature Extractor is responsible for "mining" the user characteristics, resulting three features for each user. These features are *Info Feature*, *Behavior Feature* and *Interest Feature*, which constitute the *Feature Data* in GruBa, i.e., *Feature Data* = {*Info Feature*, *Behavior Feature*, *Interest Feature*}.

3.1 Info Feature *Info Feature* employs a vector I to cover the basic info of user.

$$I = (\#R_s,\ \#E_s,\ R_{ee},\ \#B_s,\ R_{oc},\ U_t) \tag{3.1}$$

where each variable is illustrated in Table 1. Specifically, we use $\#(B_s|R(B) == 1)$ and $\#(B_s|R(B) == 0)$ to represent the number of retweeted blogs and blogs that are originally created by the user.

Table 1: Illustration of Variables in Info Feature

variable	illustration
$\#R_s$	number of followers
$\#E_s$	number of followees
R_{ee}	a ratio defined as $\frac{\#R_s}{\#E_s}$
$\#B_s$	number of blogs owned by the user
R_{oc}	a ratio defined as $\frac{\#(B_s\|R(B)==1)}{\#(B_s\|R(B)==0)}$
U_t	user type (as detailed in Table 2)

Table 2: Category of Info

value	illustration
0	$\#E_s \leq 50$ & $\#R_s \leq 50$
1	$\frac{\#E_s}{\#R_s} \geq 5$
2	$\frac{\#R_s}{\#E_s} \geq 5$
3	other cases

3.2 Behavior Feature Unlike *Info Feature*, *Behavior Feature* shows several statistics regarding the user's retweeting behavior. Such statistics include:

- a value showing the average number of retweeted blogs per week: $\#W_r$
- a normalized vector regarding the time distribution of a user's retweeting behavior: $P_t = (p'_0,\ p'_1,\ ...,\ p'_{11})$, where p'_0 is the probability that the retweeting activity happens from 0am to 2am, p'_1 is the probability that the retweeting activity happens from 2am to 4am, and so on.
- a normalized vector with respect to the gap distribution of a user's retweeting behavior: $P_g = (p''_0,\ p''_1,\ ...,\ p''_5)$, in which p''_0 is the probability that the gap between two retweeted blogs is within 1 min. Ditto for p''_1 (1 min to 1 hour), p''_2 (1 to 12 hours), p''_3 (12 to 24 hours), p''_4 (24 to 48 hours) and p''_5 (more than 48 hours).

Hence, *Behavior Feature* H per user comes with:

$$H = (\#W_r,\ P_t,\ P_g) \tag{3.2}$$

where $\#W_r$, P_t and P_g are illustrated as above.

3.3 Interest Feature Different from the straightforward notions of *Info Feature* and *Behavior Feature*, *Interest Feature* involves a process of labeling users by their interested topics. In short, with a given lexicon consisting of several *topics*, the interest feature of a user is a normalized vector, in which each dimension refers to the probability that the said user matches a *topic*.

DEFINITION 3. *A lexicon $L = \{t\}$ consists of a set of topics while each topic $t = \{c\}$ is associated with a list of cell words $\{c\}$. Each cell word c refers to one unique word in L.*

DEFINITION 4. *With a given user u, each blog $b \in B_s(u)$ could be decomposed into a set of words $\{w\}$, i.e., $b = \{w\}$.*

DEFINITION 5. *The interest feature of a given user is a normalized vector*

$$P_f = (p_0,\ p_1,\ ...,\ p_{x-1}) \tag{3.3}$$

in which the said user matches x topics in lexicon and p_i refers to the similarity of the user and each matched topic (interest). The definition of such similarity shall be detailed in each scenario (explicit/implicit interest analysis, towards words/topics, etc).

In GruBa, a word, either in the form of c or w, acts as the minimum unit for analysis. Hence, the similarity of a word pair $(w,\ c)$, i.e., $sim(w, c)$, could be generalized to the similarity of a blog against one topic $sim(b, t)$, and finally to a user versus each topic in lexicon $sim(u, \{t\})$; topics with similarity satisfying certain thresholds are allocated to the user u and constitute the interests of u.

For instance, the following steps depict the "mining" process of the interest feature P_f for a user u.

Step 1: Each blog of u is decomposed into a word set, i.e., $b = \{w\}$ where $b \in B_s(u)$.

Step 2: Explicit interests are explored. Specifically, every word w is sent to match each cell word c of lexicon topics. If w and c are identical, $sim(w, c) = 1$. Otherwise, $sim(w, c) = 0$. As to the similarity of b against a lexicon topic t, it is:

$$sim(b, t) = \sum_{i,j} sim(w_i, c_j) \tag{3.4}$$

where $sim(w_i, c_j)$ refers to the similarity of a word pair.

If $sim(b, t)$ satisfies a certain threshold, topic t is labeled to blog b; the user u is then discovered having an explicit interest (topic) t. Thus, by looking into the similarity of b against all topics in lexicon, the explicit interests of u is returned, in the form of interest feature (see Definition 5).

If none of $sim(b, t)$ could meet the threshold, i.e., explicit interest discovery over user u fails, go to *Step 3* and *Step 4* in parallel, so as to "mine" the implicit interests of u.

Step 3: A metric *TF-IDF weight* W_f is computed, i.e., employing TF-IDF [reference] to calculate the weight distribution of words in blog b:

$$W_f = \{(w_i, p_i)\} \tag{3.5}$$

where w_i refers to a single word, of which the weight is p_i, with $\sum_i p_i = 1$.

To compute such weight p_i for word w_i, a metric p''_i is first calculated as:

$$p''_i = \frac{|b_i|}{|b|} * log(\frac{|D_i|}{|D|}) \tag{3.6}$$

in which we use the operator $|\ |$ to measure the cardinality, such that $|b_i|$ is the occurrences of word w_i in blog b and $|b|$ the total occurrences of all words in b. Ditto for $|D_i|$ and $|D|$, except that the scope is the overall dataset, rather than a single blog b.

Hence, each word w_i shall get an initial weight of p_i'', upon which the normalization is performed and p_i is obtained, resulting the *TF-IDF weight* (see Definition 3.5). Go to *Step 5*.

Step 4: Similarly, another metric *Twitter-LDA weight* W_w is obtained, i.e., using Twitter-LDA [reference] to result the word weight distribution of blog b. Unlike TF-IDF [reference], Twitter-LDA [reference] first trains the overall blogs, allocating each blog with a *tag*. The structure of *tag* is as follows:

$$W_t = \{(w_i', p_i')\} \tag{3.7}$$

where w_i' refers to a word in *tag* W_t, and p_i' is the probability that w_i' appears in blogs with the said *tag*, with $\sum_i p_i' = 1$. Subsequently, W_t are leveraged to conclude W_w, i.e., $W_w = W_t$, which shares the format with that of W_f. Go to *Step 6*.

Step 5: TF-IDF [reference] based similarity is calculated. For example, the similarity (in the form of a value) of W_f over a single topic t in lexicon, written as $sim(W_f, t)$, is defined as:

$$sim(W_f, t) = \sum_i p_i * sim(w_i, t) \tag{3.8}$$

where $W_f = \{(w_i, p_i)\}$, $t = \{c_j\}$, and:

$$sim(w_i, t) = \sum_j sim(w_i, c_j) \tag{3.9}$$

Go to *Step 7*.

Step 6: Accordingly, Twitter-LDA [reference] based similarity is available. Again, a single topic t in lexicon is used for yardstick and the similarity of W_w over t, written as $sim(W_w, t)$, is defined as:

$$sim(W_w, t) = \sum_i p_i' * sim(w_i', t) \tag{3.10}$$

where $W_w = \{(w_i', p_i')\}$, $t = \{c_j\}$, and:

$$sim(w_i', t) = \sum_j sim(w_i', c_j) \tag{3.11}$$

Go to *Step 7*.

Step 7: Hence, the similarity of a blog b against a lexicon topic t is given by:

$$sim(b, t) = \alpha * sim(W_f, t) + (1 - \alpha) * sim(W_t, t) \tag{3.12}$$

where the α is a parameter by which GruBa could set flexible priorities between TF-IDF [reference] and Twitter-LDA [reference]. Go to *Step 8*.

Step 8: Repeat the above steps (Step 1 to Step 7) for the blog b over every topic in lexicon, i.e., $\forall t_k \in L$ results one similarity value of $sim(b, t_k)$. Such computation further extends to all the blogs owned by user u, such that: $\forall b_m \in B_s(u)$, $\forall t_k \in L$, there exists a similarity of $sim(b_m, t_k)$. Hence, the overall similarity of user u over lexicon topics $\{t\}$ (i.e., L), written as $S(u,\ L)$, could be denoted by a vector:

$$S(u, L) = (s_0,\ s_1,\ ...,\ s_{n-1}) \tag{3.13}$$

where n refers to the cardinality of L (i.e., number of topics in L) and s_k is the overall similarity of user u over topic t_k, which is given by:

$$s_k = \sum_m sim(b_m, t_k) \tag{3.14}$$

Among the n dimensions of $S(u,\ L)$, those with top x similarity values are selected to label the implicit interests of user u, which results an x dimensional vector P_f as described in Definition 5. Similarly, interest features of all users are returned.

As a result, the *Feature Data* for every user u, written as $F(u)$, is given by:

$$F(u) = (I,\ H,\ P_f) \tag{3.15}$$

where I, H and P_f refer to *Info Feature*, *Behavior Feature* and *Interest Feature* separately (see formulas 3.1, 3.2 and 5). And it could be written as a vector:

$$F(u) = (\#R_s, \#E_s, R_{ee}, \#B_s, R_{oc}, U_t, \#W_r, P_t, P_g, P_f) \tag{3.16}$$

where each dimension refers to a data item of GruBa.

4 User Clusterer

Providing the *Feature Data*, User Clusterer takes the charge of grouping each user concerned into a proper cluster. Algorithm 1 illustrates such overall procedure.

The idea is to enumerate a number of clustering trials (line 4) and select the optimal solution with the best coefficient value (v in line 14). In principle, each trial (referred by t in line 4) first performs a clustering task (line 5; to be detailed in section 4.1), resulting a cluster (by $l(u)$) for each user u (line 6); then, each user obtains a coefficient value $v(u)$ stemmed from the in/out-cluster distances (lines 8–10; shall be illustrated in section 4.2); finally, the averaged coefficient value of all users serves as the coefficient value of the current trial, written as $v(t)$ (line 12), by which the said selection process is conducted (line 14).

Algorithm 1 User Clustering in GruBa

Input: *Feature Data* of users $\{F(u)\}$, the minimum/maximum number of clusters N_i and N_a
Output: Optimal user clustering result R

for all $t \in [N_i, N_a]$ **do**
  group users $\{u\}$ into t clusters by $\{F(u)\}$
  clustering result $R'(t) = \{(u,\ l(u))\}$ with cluster info $l(u)$ for each user u
  for all $u \in \{u\}$ **do**
    in-cluster distance $d_i(u)$
    out-cluster distance $d_o(u)$
    coefficient value $v(u) = \frac{(d_o - d_i)}{max(d_o,\ d_i)}$
  end for
  $v(t) = Avg\{v(u)\}$
end for
if $v(a) == Max\{v(t)\}$ **then**
  $R = R'(a)$
end if
return R

Next, we shall now first detail how GruBa performs the clustering task and subsequently illustrate the computation regarding the metric of coefficient value.

4.1 Clustering in GruBa In GruBa, the clustering rests on an optimized K-Prototype [reference] algorithm, named K-Gru in this work. Similar as K-Prototype, K-Gru randomly selects the cluster kernels among samples and employs the minimum distance between them to determine an initial result, upon which the clustering tasks are iterated until the results are stable.

Unlike K-Prototype that supports vector samples in which each dimension is of numerical/categorical, K-Gru could also handle the case where a dimension is one normalized vector. Recall the sample data for User Clusterer, i.e., *Feature Data* in form of vectors (see formula 3.16), of which the data type regarding each dimension is shown as Table 3.

Table 3: Types of Dimensional Data in *Feature Data* Vector

type	data dimensions
numerical data	$\#R_s$, $\#E_s$, R_{ee}, $\#B_s$, R_{oc}, $\#W_r$
categorical data	U_t
normalized vectors	P_t, P_g, P_f

As aforementioned, the clustering of K-Gru rests on the distance between vector samples, where the dimensions are combined with numbers, categories and normalized vectors. For simplicity, we shall first illustrate the distance calculation of the simple vectors with mono data type on each dimension and then demonstrate that of complex vectors in K-Gru.

Given two numerical vectors $Y' = (y'_0, y'_1, ...)$ and $Z' = (z'_0, z'_1, ...)$, the O's Distance [reference] between Y' and Z' is given by :

$$D_n(Y', Z') = \sum_e (y_e - z_e)^2 \tag{4.17}$$

As to the categorical vectors $Y'' = (y''_0, y''_1, ...)$ and $Z'' = (z''_0, z''_1, ...)$, the H's Distance [reference] of Y'' and Z'' is:

$$D_h(Y'', Z'') = \sum_e H_e \tag{4.18}$$

where H_e refers to the H's Distance over each dimension, with $H_e = 1$ if y''_e and z''_e share the identical value, and $H_e = 0$ otherwise.

Regarding two vectors where each dimension is a normalized vector per se, Cosine Similarity [reference] is leveraged to compute the distance. Then, the distance between such two vectors $Y^* = (Y^*_0, Y^*_1, ...)$ and $Z^* = (Z^*_0, Z^*_1, ...)$ is:

$$D_v(Y^*, Z^*) = \sum_e Y^*_e \cdot Z^*_e \tag{4.19}$$

where $\cdot$ refers to the dot product operation between two normalized vectors Y^*_e and Z^*_e.

Hence, the said distance regarding the complex vectors ($Y = (Y_0, Y_1, ...)$ and $Z = (Z_0, Z_1, ...)$) in K-Gru, named G's Distance, could be deduced as:

$$D_g(Y, Z) = \sum_e G_e \tag{4.20}$$

where the distance on each dimension G_e is given by:

$$G_e = \begin{cases} (Y_e - Z_e)^2 & \text{if } Y_e/Z_e \text{ is numerical} \\ H_e \ (1 \ or \ 0) & \text{if } Y_e/Z_e \text{ is categorical} \\ Y_e \cdot Z_e & \text{if } Y_e/Z_e \text{ is of normalized vector} \end{cases} \tag{4.21}$$

4.2 Coefficient Metric Computation In GruBa, coefficient value serves as the fundamental criteria for the optimal clustering selection. Providing a clustering result, each user is associated with a cluster. For a given user u of cluster l, we employ the vector Y to denote the *Feature Data* as in formula 3.16.

DEFINITION 6. *The in-cluster distance $d_i(u)$ is the average distance to all the other users in the same cluster, i.e., $\forall u'' \in l \ \& \ u \neq u''$:*

$$d_i(u) = Avg\{D_g(Y_u, Y''_u)\} \tag{4.22}$$

DEFINITION 7. *The out-cluster distance $d_o(u)$ is measured as the minimum of the distances $\{d^*\}$ between u and other clusters ($\forall l' \neq l$), i.e.:*

$$d_o(u) = Min\{d^*(u, l')\} \tag{4.23}$$

where d^* is given by:

$$d^*(u, l') = Avg\{D_g(Y_u, Y'_u)\}\ \forall u' \in l' \tag{4.24}$$

DEFINITION 8. *The coefficient value $v(u)$ is thus concluded:*

$$v(u) = \frac{(d_o - d_i)}{max(d_o,\ d_i)} \tag{4.25}$$

Intuitively, a good clustering solution should result bigger d_o and smaller d_i, such that samples with obvious differences go to various clusters and vice versa. When d_o is far more than d_i, coefficient value approaches to 1. Hence, the larger coefficient value is, the better clustering performs, by which the optimal solution is selected.

5 Group Modeler

Recall the central problem of GruBa, where the retweeting behaviors of users are modeled. Specifically, such model is built by Group Modeler for each user group and thus named as group model. To avoid ambiguity, we shall use the term of *items* to denote the data for training the group model. A given *item* is either positive or negative.

DEFINITION 9. *An item E involves a blog b and a user f such that $f \in R_s(O(b))$, i.e., f is a follower of b's owner.*

$$E \in \begin{cases} positive\ items & if\ f\ retweeted\ b \\ negative\ items & if\ f\ did\ not\ retweet\ b \end{cases} \tag{5.26}$$

And the data of item E could be further divided into three parts.

- *User Part* contains a list of aforementioned metrics $\{\#R_s, \#E_s, R_{ee}, \#B_s, \#W_r\}$.
- *Blog Part* consists of a metric C_h, referring to the correlation between blog contents and recent events returned by Ring [reference]. C_h is in the form of a normalized vector with each dimension represents one event (similar as P_f in formula 3.3). Specifically, each event could be viewed as a topic t, over which the correlation of a blog b could be obtained by formula 3.12.
- *Interaction Part* includes three correlation metrics. They are of blog b versus the user u's *Interest Feature* $P_f(u)$ (a.k.a. long-term/stable interest in this work), b versus u's short-term interest $P_s(u)$ that is mined from u's recent blogs (e.g., within 30 days) in the same manner of $P_f(u)$, and b's timestamp versus the time distribution of u's retweeting behavior P_t.

As a result, the obtained group model could learn what does a positive/negative item look like over each metric mentioned above.