Tripartite Graph Clustering for Dynamic Sentiment Analysis on Social Media (SIGMOD'2014)

Linhong Zhu, Aram Galstyan, James Cheng and Kristina Lerman
Outline

- Introduction
- Background & Challenges
- Contributions
- Problem definition
- Offline framework
- Online framework
- Experiments
Introduction
Introduction

- There has been a significant growth in the use of social media platforms such as Twitter.

- Many people are seeking ways to analyze the sentiments of users through the Twitter platform on their products, services and policies.
Introduction

User-level
Stances to GMO labeling
- Neu
- Pos
- Neg

Tweet-level
- $p_1$: Should India go back to poverty when it used little of ag technologies like #GMOs
- $p_2$: Support the #California #GMO Labeling Ballot Initiative #prop37
- $p_3$: Monsanto is pure evil
- $p_4$: Ah ha! Love this Yes on #Prop37 add :) 
- $p_5$: GM crops increased farm incomes worldwide by $14$ billion in 2010!!!
- $p_6$: GM crops poses no greater risk than conventional food

Polarity
- Neg
- Pos
- Pos
- Neg
- Neu

Figure 1: An example of both user-level and tweet-level sentiments toward the topic "labeling genetically modified organism (GMO)"
Background & Challenges
Most prior work on Twitter sentiment analysis has focused on:

- understanding the sentiments of individual tweets
- understanding the user-level sentiments
- both tweet-level and user-level sentiments[1]

Short texts such as tweets will generally be very noisy and error prone

(Bob is positive towards GMO labeling, however, tweet p3 might be classified as “negative” due to the occurrence of word “evil” and simple aggregation of both p3 and p4 would produce incorrect sentiment for Bob.)

In contrast, inferring the sentiment of p3 jointly with the sentiment of Bob’s other tweets can potentially produce a more accurate classification for tweet p3 and Bob’s overall sentiment.

This example motivates us to jointly analyze both tweet-level sentiments and user-level sentiments, by modeling the dependencies among users, tweets and words.
Another important challenge is understanding and characterizing the temporal evolution of user-level sentiment. (For example, user Adam in Figure 1 seems to be against GMO labeling at first, but changes his mind in support of GMO labeling, perhaps due to interactions with other users.)

This example motivates us to develop dynamic sentiments analyse method.
Contributions
Contributions

- To address the above-mentioned challenges, we study the sentiment co-clustering problem, which aims to simultaneously cluster the sentiment of tweets and users.

- We propose a unified unsupervised tri-clustering framework, to solve the sentiment-clustering problem by solving its dual problem: co-cluster a tripartite graph which represents dependencies among tweets, users and features into sentiment class.

- We give an example of a tripartite graph co-clustering as follows.
Figure 2: Co-clustering of a tripartite graph of features, users and tweets. The dashed/solid lines represent posting/re-tweeting relations between users and tweets.

Example 1  Figure 2 shows an example of a tripartite graph, which models the correlation among features (e.g., words), tweets and users. There are three layers of nodes (i.e., features $F$, users $U$, and tweets $P$), where a feature node $f \in F$ is connected with a tweet $p \in P$ if the tweet $p$ contains the feature $f$; and a user $u \in U$ is connected with a tweet $p$ if either $u$ posts or re-tweets the tweet $p$. Therefore, if we can obtain a good clustering over the tripartite graph, for instance, we obtain three subsets $\{f_1, f_2, f_3, p_1, p_2, p_3, u_1, u_2, u_3\}$, $\{f_4, f_5, p_4, u_4\}$ and the remaining, then the users are clustered into positive users$\{u_1, u_2, u_3\}$, neutral users$\{u_4\}$ and negative users$\{u_5, u_6\}$, while the tweets are clustered into three subsets $\{p_1, p_2, p_3\}$, $\{p_4\}$ and $\{p_5, p_6, p_7\}$ simultaneously.
Contributions

- This example reveals several advantages of the proposed tri-clustering framework.

- First, triclustering framework exploits the duality between sentiment clustering and tripartite graph co-clustering to perform both user-level and tweet-level sentiment analysis.

- Second, since co-clustering is an unsupervised approach, neither labeled data nor high quality of labels are required, though performance can be improved by including high quality labeled data or outputs of other sentiment analysis approaches.

- Finally, a useful feature of co-clustering is that it can utilize the intermediate clustering results of tweets to improve the clustering results of users, and vice versa.

- We extend the tripartite graph co-clustering to an online setting.
Problem definition
We first introduce the notations used in this paper, which are listed in Table 1 for quick reference.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Explanations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$ and $m$</td>
<td>number of tweets and users</td>
</tr>
<tr>
<td>$l$ and $k$</td>
<td>number of features and clusters</td>
</tr>
<tr>
<td>$P$, $U$ and $F$</td>
<td>a set of tweets, users, and features</td>
</tr>
<tr>
<td>$X_p/X_u$</td>
<td>tweet-feature/user-feature matrix</td>
</tr>
<tr>
<td>$X_r/G_u$</td>
<td>user-tweet/user-user matrix</td>
</tr>
<tr>
<td>$S_p/S_f/S_u$</td>
<td>tweet/feature/user cluster matrix</td>
</tr>
<tr>
<td>$H_p/H_u$</td>
<td>$k \times k$ association matrix</td>
</tr>
<tr>
<td>$M_{(i)}$</td>
<td>the $i^{th}$ row of matrix $M$</td>
</tr>
<tr>
<td>tr$(M)/</td>
<td></td>
</tr>
<tr>
<td>$D_u/L_u$</td>
<td>Diagonal/Laplacian matrix of $G_u$</td>
</tr>
<tr>
<td>$M(t)$</td>
<td>matrix $M$ at timestamp $t$</td>
</tr>
<tr>
<td>$n(t)/m(t)$</td>
<td>number of tweets/users at time $t$</td>
</tr>
<tr>
<td>$X_{ud}/X_{un}$</td>
<td>user-feature matrix for disappeared users, new users, and evolving users</td>
</tr>
</tbody>
</table>

With the terminologies defined above, we now formally define our sentiment co-clustering problem as follows:

**Problem 1** Given a series of temporal tweet data $\{p_1, p_2, \cdots, p_t\}$ that are related to the same topic, our purpose is to automatically and collectively infer the sentiments of all the observed tweets $S_p \in \mathbb{R}^{n \times k}_+$, and the temporal sentiments of all the observed users $\{S_u(1) \in \mathbb{R}^{m \times k}_+, S_u(2), \cdots, S_u(t)\}$. 
Offline framework
Figure 3 illustrates the overall idea of the two-level clustering in our framework.

Figure 3: Offline tri-clustering framework overview (the figure is best viewed in color)
Offline framework

- We employ a two-level clustering for a given tripartite graph.

- **First**, at the intra-level, a tripartite graph is separated into three mutually related bipartite graphs,
  1) tweet-feature bipartite graph (matrix representation $X_p \in \mathbb{R}_{+}^{n \times l}$)
  2) user-feature bipartite graph (matrix representation $X_u \in \mathbb{R}_{+}^{m \times l}$)
  3) user-tweet bipartite graph (matrix representation $X_r \in \mathbb{R}_{+}^{m \times n}$)

- **Then**, at the inter-level, we deal with the mutual dependency among bipartite graphs, and show that the intermediate clustering results of a bipartite graph can be further applied to the clustering process of another bipartite graph.

- **Finally**, inter-level clustering and intra-level clustering are mutually performed.

At the intra-level, we study the clustering for each single bipartite graph, and then formulate the bipartite graph clustering as a factorization problem for the non-negative matrix (NMF) representation of the given bipartite graph.
Offline framework

With the idea of the two-level clustering, now we propose a new Tri-clustering framework to perform both user-level and tweet-level sentiment analysis. The objective of our Triclustering framework is formulated as follows:

\[
\begin{align*}
\arg \min_{S_f, S_u, S_p, H_u, H_p \geq 0} & \left\{ \|X_p - S_p H_p S_f^T\|_F^2 + \|X_u - S_u H_u S_f^T\|_F^2 \right. \\
& + \|X_r - S_u S_p^T\|_F^2 \\
& + \alpha \|S_f - S_{f0}\|_F^2 + \beta \text{tr}(S_u^T L_u S_u) \right\} \\
\text{s.t.} & \quad S_f S_f^T = I, \ S_p S_p^T = I, \ S_u S_u^T = I
\end{align*}
\]

first three terms represents the intra-level bipartite graph clustering, and the aggregation of these three terms represents the inter-level bipartite graph clustering.

In addition, we also incorporate sentiment lexicon information (on the top of Figure 3) into the framework by adding regularization functions to features, and emotion correlation between users and retweeting users by using user-graph regularization (see right bottom of Figure 3).
Offline framework

- In the following, we elaborate more details about each component.

**co-clustering of tweets and features.**

$$\min_{S_f, H_p, S_p} \|X_p - S_p H_p S_f^T\|^2_F$$  \hspace{1cm} (2)

where $\|M\|_F$ denotes the Frobenius norm of a matrix $M$, $S_f \in \mathbb{R}^{l \times k}$ denotes the feature cluster information with $S_f(i,j)$ represents the probability that the $i$-th feature belongs to the $j$-th cluster, $H_p$ represents the association between features and tweet classes.

**co-clustering of users and features.**

$$\min_{S_f, H_u, S_u} \|X_u - S_u H_u S_f^T\|^2_F$$  \hspace{1cm} (3)

where $H_u$ denotes the association between features and user classes. This is similar to tweet clustering: we argue that users can be characterized by the word features of their tweets and word features can be clustered according to their distribution among users.
Offline framework

- In the following, we elaborate more details about each component.

**co-clustering of users and features.**

\[
\min_{S_f, H_u, S_u} \| X_u - S_u H_u S_f^T \|_F^2
\]  

(3)

where \( H_u \) denotes the association between features and user classes. This is similar to tweet clustering: we argue that users can be characterized by the word features of their tweets and word features can be clustered according to their distribution among users.

**co-clustering of re-tweeting users and tweets**

\[
\min_{S_u, S_p} \| X_r - S_u S_p^T \|_F^2
\]  

(4)

where \( X_r \) is the user-retweet matrix and \( X_r(i,j) \) represents that the \( i \)-th user retweets the \( j \)-th tweet.
Offline framework

- In the following, we elaborate more details about each component.

emotion consistence between clusters and sentiment classes.

$$\min_{S_f} \| S_f - S_{f0} \|^2_F$$  \hspace{1cm} (5)

where $S_{f0}$ represents the sentiment information of features (e.g. sentiment lexicon), and $S_{f0(i,j)}$ is the probability that the $i$-th feature belongs to the $j$-th sentiment class. In this component, we add a regularization to make the feature representation more relevant to the task of sentiment representation, and the clusters close to the sentiment classes.

emotion correlation between users and re-tweeting users.

$$\min_{S_u, G_u} \frac{1}{2} \sum_i \sum_j \| S_u(i) - S_u(j) \|_2^2 G_{u(i,j)}$$

$$= \text{tr}(S_u^T L_u S_u)$$  \hspace{1cm} (6)

where $G_u$ is a user-user graph of which each node is a user and each edge denotes the user-user re-tweeting relationship, $S_u(i)$ is a vector which represents the cluster association for user $i$, $L_u = D_u - G_u$ is the Laplacian matrix of the user-user re-tweeting graph, and $\text{tr}$ represents the trace of a matrix. This equation incurs a penalty if two users are close in the user-user graph but have different sentiment labels.
Offline Optimization Algorithm

- Updating Sf
- Updating Sp
- Updating Su
- Updating Hp
- Updating Hu
Updating Sf

- Updating Sf. Optimizing Eq. (1) with respect to Sf is equivalent to solving

\[
\begin{align*}
\min_{S_f \geq 0} & \quad \|X_p - S_p H_p S_f^T\|^2_F + \|X_u - S_u H_u S_f^T\|^2_F + \alpha \|S_f - S_{f0}\|^2_F \\
\text{subject to} & \quad S_f S_f^T = I
\end{align*}
\]

We introduce the Largrangian multiplier \( \mathcal{L} \) for non-negative constraint (i.e., \( S_f \geq 0 \)) and \( \Delta \) for orthogonal constraint (i.e., \( S_f S_f^T = I \)) to \( S_f \) in Eq. (1), which leads to the following Largrangian function \( L(S_f) \):

\[
L(S_f) = \|X_p - S_p H_p S_f^T\|^2_F + \|X_u - S_u H_u S_f^T\|^2_F + \alpha \|S_f - S_{f0}\|^2_F - \text{tr}[\mathcal{L}_{S_f} \cdot S_f^T] + \text{tr}[\Delta S_f (S_f S_f^T - I)]
\]

The next step is to optimize the above terms w.r.t. \( S_f \). [We set \( \frac{\partial L(S_f)}{\partial S_f} = 0 \), and obtain:

\[
\mathcal{L}_{S_f} = -2X_p^T S_p H_p + 2S_f H_p^T S_p^T S_p H_p - 2X_u^T S_u H_u + 2S_f H_u^T S_u S_u H_u + 2\alpha (S_f - S_{f0}) + 2S_f \Delta S_f
\]
Updating Sf

Using the KKT condition $\mathcal{L}_{S_f}(i, j) \cdot S_f(i, j) = 0$ [19], we obtain:

$$[-X_P^T S_p H_p + S_f H_P^T S_p S_p H_p - X_u^T S_u H_u + S_f H_u^T S_u S_u H_u + \alpha(S_f - S_{f0}) + S_f \Delta_{S_f}(i, j)] S_f(i, j) = 0$$

where $\Delta_{S_f} = S_f^T X_u^T S_u H_u - H_u^T S_u S_u H_u + S_f^T X_p^T S_p H_p - H_p^T S_p S_p H_p - \alpha S_f^T (S_f - S_{f0})$.

Following the updating rules proposed and proved in [1], we have:

$$S_f \leftarrow S_f \circ \sqrt{\frac{X_u^T S_u H_u + X_p^T S_p H_p + \alpha S_{f0} + S_f \Delta_{S_f}^-}{S_f H_u^T S_u S_u H_u + S_f H_p^T S_p S_p H_p + \alpha S_f + S_f \Delta_{S_f}^+}}$$

(7)

where $\Delta_{S_f}^+ = (|\Delta_{S_f}| + \Delta_{S_f})/2$ and $\Delta_{S_f}^- = (|\Delta_{S_f}| - \Delta_{S_f})/2$.


Updating Sp

- Updating Sp. Optimizing the objective function in Eq. (1) with respect to $S_p$ is equivalent to solving

$$\min_{S_p \geq 0} \|X_p - S_p H_p S_f^T\|_F^2 + \|X_r - S_u S_p^T\|_F^2$$

subject to $S_p S_p^T = I$

Thus, we introduce the Lagrangian multiplier $L$ for non-negative constraint (i.e., $S_p \geq 0$) and $\Delta$ for orthogonal constraint (i.e., $S_p S_p^T = I$) to $S_p$ in the above term, which leads to the following Lagrangian function $L(S_p)$:

$$L(S_p) = \|X_p - S_p H_p S_f^T\|_F^2 + \|X_r - S_u S_p^T\|_F^2$$
$$- \text{tr}[\mathcal{L}_{S_p} S_p^T] + \text{tr}[\Delta_S (S_p S_p^T - I)]$$

The next step is to optimize the above terms w.r.t $S_p$. We set $\frac{\partial L(S_p)}{\partial S_p} = 0$, and then use the Frobenius norm of a matrix $\|M\|_F^2 = \text{tr}(M^T M)$, we get:

$$\mathcal{L}_{S_p} = -2X_p S_f H_p^T + 2S_p H_p S_f^T S_f H_p^T$$
$$-2X_r^T S_u + 2S_p S_u^T S_u + 2S_p \Delta_S$$
Updating Sp

Consider the KKT condition $\mathcal{L}_{S_p}(i, j) \cdot S_p(i, j) = 0$ \[19\], we have:

$$[-2X_p S_f H_p^T + 2S_p H_p S_f^T S_f H_p^T - 2X_r^T S_u + 2S_p S_u^T S_u + 2S_p \Delta S_p](i, j) \cdot S_p(i, j) = 0$$

where $\Delta S_p = S_p^T X_p S_f H_p^T - H_p S_f^T S_f H_p^T + S_p^T X_r^T S_u - S_u^T S_u$. Let $\Delta^+_{S_p} = (|\Delta S_p| + \Delta S_p)/2$ and $\Delta^-_{S_p} = (|\Delta S_p| - \Delta S_p)/2$, we get:

$$[-(X_p S_f H_p^T + X_r^T S_u + S_p \Delta^-_{S_p}) + (S_p H_p S_f^T S_f H_p^T + S_p S_u^T S_u + S_p \Delta^+_{S_p})](i, j) \cdot S_p(i, j) = 0$$

Following the updating rules proposed and proofed by \[9\], we have the updating rule of $S_p$:

$$S_p \leftarrow S_p \circ \sqrt{\frac{X_p S_f H_p^T + X_r^T S_u + S_p \Delta^-_{S_p}}{S_p H_p S_f^T S_f H_p^T + S_p S_u^T S_u + S_p \Delta^+_{S_p}}}$$

(9)
Updating Summary

- Updating $S_f$
  \[ S_f \leftarrow S_f \circ \sqrt{\frac{X_u^T S_u H_u + X_p^T S_p H_p + \alpha S_f 0 + S_f \Delta S_f^-}{S_f H_u^T S_u H_u + S_f H_p^T S_p H_p + \alpha S_f + S_f \Delta S_f^+}} \]  

- Updating $S_p$
  \[ S_p \leftarrow S_p \circ \sqrt{\frac{X_p S_f H_p^T + X_r^T S_u + S_p \Delta S_p^-}{S_p H_p^T S_f H_p^T + S_p S_u^T S_u + S_p \Delta S_p^+}} \]  

- Updating $S_u$
  \[ S_u \leftarrow S_u \circ \sqrt{\frac{X_u S_f H_u^T + X_r S_p + \beta G_u S_u + S_u \Delta S_u^-}{S_u H_u^T S_f H_u^T + S_u S_p^T S_p + \beta D_u S_u + S_u \Delta S_u^+}} \]  

- Updating $H_p$
  \[ H_p \leftarrow H_p \circ \sqrt{\frac{S_p^T X_p S_f}{S_p^T S_p H_p S_f^T}} \]  

- Updating $H_u$
  \[ H_u \leftarrow H_u \circ \sqrt{\frac{S_u^T X_u S_f}{S_u^T S_u H_u S_f^T}} \]
Offline framework

Algorithm 1 offline algorithm for tri-clustering

Input: $X_p$, $X_r$, $X_u$, $S_{f0}$, user-user retweeting graph $G_u$, parameters: $\alpha$ and $\beta$

Output: $S_u$, $S_p$, $S_f$

1: initialize $S_u$, $S_p$, $S_f$, $H_u$, $H_p \geq 0$
2: while not converge:
3:     update $S_p$ according to Eq. (9);
4:     update $H_p$ according to Eq. (12);
5:     update $S_u$ and $H_u$ according to Eq. (11) and Eq. (13);
6:     update $S_f$ according to Eq. (7);
7: return $S_u$, $S_p$, $S_f$
Online framework
Online framework

- We have introduced the offline Tri-clustering framework to handle static data. We now present our online framework, where the temporal data is coming in a streaming fashion.

There are two naive ways to deal with temporal data:

- (1) applying the offline tri-clustering framework to the entire dataset whenever new data is added

- (2) applying tri-clustering only to new data independently at each time interval.
Online framework

- Our online framework achieves a good tradeoff between the above two extremes and is able to study the evolution of sentiments.

- Before we present the online framework, we first introduce the following two observations:

  1. The frequency distribution of vocabularies changes over time; however, the sentiments of vocabularies do not change or change slowly over time.

  2. Considering the entire population, the majority of users rarely change their mind within a short time.
• To verify Observation 1, we plot the frequency of vocabularies used by the same user in two different time periods in our dataset and the top-8 words with the highest frequency in each pos/neg class from our dataset.

Table 2: Top-8 words with highest frequency.

<table>
<thead>
<tr>
<th></th>
<th>Neg</th>
<th>Pos</th>
</tr>
</thead>
<tbody>
<tr>
<td>corn</td>
<td>(1463)</td>
<td>yeson37</td>
</tr>
<tr>
<td>farmer</td>
<td>(1223)</td>
<td>labelgmo</td>
</tr>
<tr>
<td>noprop37</td>
<td>(1211)</td>
<td>monsanto</td>
</tr>
<tr>
<td>crop</td>
<td>(881)</td>
<td>stopmonsanto</td>
</tr>
<tr>
<td>million</td>
<td>(778)</td>
<td>earighttoknow</td>
</tr>
<tr>
<td>feed</td>
<td>(380)</td>
<td>health</td>
</tr>
<tr>
<td>India</td>
<td>(380)</td>
<td>safe</td>
</tr>
<tr>
<td>seed</td>
<td>(355)</td>
<td>cancer</td>
</tr>
</tbody>
</table>

Figure 4: The evolution of features

This observation provides us the intuition to utilize the previous sentiment clustering results of features to improve the clustering quality of tweets/users.
Observation 2, has been shown in various existing works.

[1] showed that the sentiments of users before election are highly correlated with the sentiments of users after election with Pearson correlated coefficient of 0.851.

[2] That for several topics, two posts created by the same user have similar sentiments.

This observation motivates us to improve the clustering quality of users based on their previous clustering results.


The two observations tell us:

According to Observation 1: Sentiment information can be naturally obtained through factorizing new data matrix $X_p(t)$, with reference to few previous sentiment clustering results of features.

we group users at time $t$ into three categories: new users, disappeared users, and evolving users.

According to Observation 2:
For new users, we conduct sentiment analysis based on the current user and tweet information and past sentiment clustering results of features;
For evolving users, Observation 2 suggests that previous sentiment clustering results for these users are useful for current sentiment analysis.
How to make use of previous sentiment results for current sentiment analysis?
Online framework

- We adopt the **temporal regularization technique** to utilize the previous clustering results of both users and features, as suggested by the above two observations.

- We define the temporal regularization of a time-dependant matrix $M(t)$ to measure the smoothness of evolution as follows:

$$R(M(t)) = \sum_t \|M(t) - M_w(t)\|_F^2$$ (18)

where $M(t)$ denotes the matrix information at time $t$ and $M_w(t)$ denotes the past information of matrix $M$ within $[t - w, t)$. A larger value of Eq. (18) means less smoothness of evolution.

Now we address the unsolved question in Eq. (18), i.e., how to utilize the previous results for current sentiment analysis. $M_w(t)$ can be simply initialized as an aggregation of all the previous results within $[t - w, t)$. A natural modification is to give higher importance to more recent information. For example, sufficient results within $[t - w, t)$ are aggregated over time, and an exponential decay is used to forget out-of-date results. Thus, we define $S_{fw}(t) = \sum_{i=1}^{w-1} \tau^i S_f(t-i)$ and $S_{uw}(t) = \sum_{i=1}^{w-1} \tau^i S_f(t-i)$, where $\tau \in (0, 1]$ is the time-decaying factor.
We adopt the discussion above motivates the following objective function that is optimized at every time point $t$ (Figure 5 depicts the overall online framework with $w=2$):

$$
\arg \min_{S_f(t), S_u(t), S_p(t), H_u(t), H_p(t) \geq 0} \{ ||X_p(t) - S_p(t)H_p(t)S_f(t)^T||_F^2 
+ ||X_{u(e,n)}(t) - S_{u(e,n)}(t)H_u(t)S_f(t)^T||_F^2 
+ ||X_{r(e,n)}(t) - S_{u(e,n)}(t)S_p(t)^T||_F^2 
+ \alpha ||S_f(t) - S_{fw}(t)||_F^2 
+ \beta \text{tr}(S_u(t)^T L_u(t)S_u(t)) 
+ \gamma ||S_{u(d,e)}(t) - S_{uw}(t)||_F^2 \}
$$

(19)

s.t. $S_f(t)S_f(t)^T, S_p(t)S_p(t)^T, S_u(t)S_u(t)^T = I$

Figure 5: Online tri-clustering framework for dynamic sentiment clustering
Updating

- $Hu(t)$, $Hp(t)$, and new tweets $Sp(t)$

\[
Hu(t) \leftarrow H_u(t) \circ \sqrt{\frac{S_{u(e,n)}^T X_{u(e,n)} S_f}{S_{u(e,n)}^T S_{u(e,n)} H_u S_f^T S_f}}(t)
\]  

(20)

\[
Hp(t) \leftarrow H_p(t) \circ \sqrt{\frac{S_{p}^T X_{p} S_f}{S_{p}^T S_p H_p S_f^T S_f}}(t)
\]  

(21)

\[
Sp(t) \leftarrow S_p(t) \circ \sqrt{\frac{X_p S_f H_p^T + X_u^T S_u + S_p \Delta_{S_p}}{S_p H_p S_f^T S_f H_p^T + S_p S_u S_u + S_p \Delta_{S_p}^+}}(t)
\]  

(22)
Evolving Features. According to Observation 1, temporal regularization is used to ensure a smooth evolution from $S_{fw}(t)$ to $S_{f}(t)$. Therefore, for the optimization of $S_{f}(t)$, we first need to pay attention to the computation of $S_{fw}(t)$, which is the time-decaying aggregation of previous sentiment clustering results of features. Thus, we derive the update rule for $S_{f}(t)$ as follows:

$$S_{f}(t) \leftarrow S_{f}(t) \circ \sqrt{\frac{X^{T}_{u}S_{u}H_{u} + X^{T}_{p}S_{p}H_{p} + \alpha S_{fw} + S_{f} \Delta_{S_{f}}}{S_{f}H^{T}_{u}S_{u}H_{u} + S_{f}H^{T}_{p}S_{p}H_{p} + \alpha S_{f} + S_{f} \Delta^{+}_{S_{f}}}}(t)$$

(23)

where

$$\Delta_{S_{f}}(t) = S_{f}(t)^{T}X_{u}(t)^{T}S_{u}(t)H_{u}(t) - H_{u}(t)S_{u}(t)^{T}S_{u}(t)H_{u}(t) + F(t)^{T}X_{p}(t)^{T}S_{p}(t)H_{p}(t) - H_{p}(t)S_{p}(t)^{T}S_{p}(t)H_{p}(t) - \alpha S_{f}(t)^{T}(S_{f}(t) - S_{fw}(t))$$.
New Users. For a new user, we do not have the smooth constraint for temporal evolution, and the sentiment information $S_u(t)$ at time $t$ can be obtained locally by factorizing the current data matrix $X_u(t)$ and $X_r(t)$. However, those new users might be connected to existing users through re-tweeting relations, and hence the optimization of $X_u(t)$ should be performed under the temporal graph regularization as well.

\[ S_u(t) \leftarrow S_u(t) \circ \sqrt{\frac{X_u S_f H_u^T + X_r S_p + \beta G_u S_u + S_u \Delta_{S_{un}}^-(t)}{S_u H_u S_f^T S_f H_u^T + S_u S_p^T S_p + \beta D_u S_u + S_u \Delta_{S_{un}}^+(t)}} \]  

(24)

where $\Delta_{S_{un}}(t) = S_u(t)^T X_u(t) S_f(t) H_u(t)^T - H_u(t) S_f(t)^T S_f(t) H_u(t)^T + S_u(t)^T X_r(t) S_p(t) - S_p(t)^T S_p(t) - \beta S_u(t)^T L_u(t) S_u(t)$. 
Updating

Evolving Users. The computation of $S_u(t)$ for evolving users is similar to the optimization for new users except that we also have the smooth constraint for evolution (i.e., the sentiment information of users $S_u(t)$ has a steady transmission from the past $S_{uw}(t)$). In the following, we present the details about deriving updating rule for evolving users $S_u(t)$.

Optimizing the objective function in Eq. (19) with respect to existing users’ sentiments $S_u$ is equivalent to solving (for simplicity, we write $A(t)B(t)C(t)$ as $ABC(t)$):

$$
\min_{S_u \geq 0} \left\| X_u(t) - S_u S_f ^{T} (t) \right\|_F^2 + \left\| X_r(t) - S_u S_p ^{T} (t) \right\|_F^2 \\
+ \beta \text{tr} (S_u ^{T} L_u S_u (t)) + \gamma \left\| S_u (t) - S_{uw}(t) \right\|_F^2 \text{subject to } S_u S_u ^{T} (t) = I
$$

Following the updating rules proposed and proofed by [9], we have the updating rule of $S_u(t)$:

$$
S_u(t) \leftarrow S_u(t) \circ \left[ \frac{X_u S_f H_u ^{T} (t) + X_r S_p (t) + \beta G_u S_u (t) + S_u \Delta^- \tilde{S}_u (t) + \gamma S_{uw}(t)}{S_u H_u S_f ^{T} H_u ^{T} (t) + S_u S_p ^{T} S_p (t) + \beta D_u S_u (t) + S_u \Delta^+ \tilde{S}_u (t) + \gamma S_u (t)} \right]
$$

(26)
Algorithm 2 On-line algorithm for dynamic sentiment clustering

Input: New data $X_p(t)$, $X_r(t)$, $X_u(t)$, user-user re-tweeting graph $G_u(t)$, old clustering matrix $S_{fw}(t)$ and $S_{uw}(t)$, parameters: $\alpha$, $\beta$, $\gamma$, $w$ and $\tau$

Output: $S_u(t)$, $S_p(t)$, $S_f(t)$, $H_u(t)$, and $H_p(t)$

1: initialize $S_f(t) = S_{fw}(t)$, $S_{u(d,e)}(t) = S_{uw}(t)$;
2: randomly initialize $S_p(t), H_p(t), H_u(t) \geq 0$;
3: while not converge:
4: update $S_f(t)$ according to Eq. (23);
5: update $S_p(t), H_p(t)$ according to Eq. (22) and Eq. (21);
6: update $H_u(t)$ according to Eq. (20);
7: for new user: update according to Eq. (24);
8: for evolving user: update according to Eq. (26);
9: return $S_u(t), S_p(t), S_f(t)$;
Experiments
Experiments

● Dataset

We use real Twitter dataset about “California ballot initiatives” collected between August 2012 and December 2012.

Specifically, we choose two popular ballot initiatives, Propositions 30 (Temporary Taxes to Fund Education) and 37 (Genetically Engineered Foods, Labeling)

● Matrix Sf0

In addition, we use the automatically built sentiment lexicon “Yes” word lists and “No” word lists [1] to initialize the feature sentiment class matrix Sf0.

Experiments

- Evaluate the performance of tri-clustering

**Clustering Accuracy.** Given an outputted cluster $o \in C$ and with reference to a ground truth class $g \in G$, assume that we assign the outputted cluster with ground truth label using the majority vote, then the clustering accuracy of the outputted clustering $C$ on the ground truth clustering $G$ evaluates the percentage of data with correct assignments.

$$A(C, G) = \frac{1}{n} \sum_{o \in C} \max_{g \in G} |o \cap g|$$

where $n$ is the number of data samples.
**Normalized Mutual Information (NMI).** Given the outputted clustering \( C \) and ground truth clustering \( G \), the NMI is defined as:

\[
NMI(C, G) = \frac{2 \times I(C; G)}{H(C) + H(G)}
\]

where \( H(C) \) and \( H(G) \) denotes the entropy, and \( I(C; G) \) is the mutual information between \( C \) and \( G \), which is defined as

\[
I(C; G) = \frac{\sum_i \sum_j p(o_i \cap g_j) \log \frac{p(o_i \cap g_j)}{p(o_i)p(g_j)}}{\sum_i \sum_j \frac{|o_i \cap g_j|}{n} \log \frac{n|o_i \cap g_j|}{|o_i||g_j|}}
\]
Evaluate(different parameter)

User-Level

(a) Clustering accuracy

(b) NMI

Figure 6: User-level quality comparison when varying $\alpha$ and $\beta$ on Proposition 30 data (the figure is best viewed in color)

$\alpha = 0, \beta = [0.5, 0.8]$, or $\alpha = [0.7, 1], \beta = 1$

$\alpha = 0, \beta = [0.6, 0.9]$

$\alpha = 0, \beta = [0.5, 0.8]$
Evaluate(different parameter)

Tweet-Level

(a) Clustering accuracy
(b) NMI

Figure 7: Tweet-level quality comparison when varying $\alpha$ and $\beta$ on Proposition 30 data (the figure is best viewed in color)

$\alpha = 0.1, \beta = [0.8, 0.9]$
Moreover, in order to balance between the tweet-level performance and user-level performance, in all the following offline experiments, we set

\[ \alpha = 0.05, \beta = 0.8. \]
Existing Methods for Comparison(Offline)

- **Tweet-Level**
  - ESSA(u)
  - SVM(s)
  - Naive Bayes(s)
  - UserReg(ss, 10% labels)
  - Label Propagation (LP)(ss) (with 5% labels (LP-5) and 10% labels (LP-10))

- **User-Level**
  - BACG(u)
  - SVM(s)
  - Naive Bayes(s)
  - UserReg(ss, 10% labels)
  - Label Propagation (LP)(ss) (with 5% labels (LP-5) and 10% labels (LP-10))
### Existing Methods for Comparison (Offline)

**Tweet-Level**

#### Table 4: Tweet-level sentiment analysis comparison

<table>
<thead>
<tr>
<th>Metric</th>
<th>Accuracy</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prop</td>
<td>30</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supervised</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM [28]</td>
<td>89.35</td>
<td>93.17</td>
</tr>
<tr>
<td>NB [11]</td>
<td>85.75</td>
<td>89.22</td>
</tr>
<tr>
<td>Semi-supervised</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LP-5 [12, 29]</td>
<td>77.20</td>
<td>87.49</td>
</tr>
<tr>
<td>LP-10 [12, 29]</td>
<td>86.60</td>
<td>88.20</td>
</tr>
<tr>
<td>UserReg-10 [7]</td>
<td>86.76</td>
<td>90.08</td>
</tr>
<tr>
<td>Unsupervised</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESSA [15]</td>
<td>81.69</td>
<td>85.87</td>
</tr>
<tr>
<td>Tri-clustering</td>
<td>81.87</td>
<td>92.15</td>
</tr>
<tr>
<td>Online tri-clustering</td>
<td>91.88</td>
<td>92.24</td>
</tr>
</tbody>
</table>


## Existing Methods for Comparison (Offline)

### User-Level

<table>
<thead>
<tr>
<th>Metric</th>
<th>Accuracy</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prop</td>
<td>30</td>
<td>37</td>
</tr>
<tr>
<td><strong>Supervised</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM [28]</td>
<td>89.81</td>
<td>87.84</td>
</tr>
<tr>
<td>NB [11]</td>
<td>88.69</td>
<td>83.8</td>
</tr>
<tr>
<td><strong>Semi-supervised</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LP-5 [30]</td>
<td>31.77</td>
<td>82.05</td>
</tr>
<tr>
<td>LP-10 [30]</td>
<td>77.45</td>
<td>84.25</td>
</tr>
<tr>
<td>UserReg-10 [7]</td>
<td>82.10</td>
<td>84.28</td>
</tr>
<tr>
<td><strong>Unsupervised</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BACG [34]</td>
<td>75.37</td>
<td>70.51</td>
</tr>
<tr>
<td>Tri-clustering</td>
<td>86.88</td>
<td>86.17</td>
</tr>
<tr>
<td>Online tri-clustering</td>
<td>89.22</td>
<td>88.48</td>
</tr>
</tbody>
</table>

Table 5: User-level sentiment analysis comparison
Comparison(Online)

- For online framework, we compare our algorithm with two baseline algorithms, mini-batch and full-batch.

- Mini-batch algorithm, which applies our offline tri-clustering algorithm to each snapshot of new data matrices. *(high scalability but may sacrifice the quality)*

  - Full-batch algorithm, which applies the offline tri-clustering algorithm to the entire dataset whenever new data arrives at each timestamp. *(high quality but very time-consuming)*
Comparison(Online)

- For online framework, we compare our algorithm with two baseline algorithms, mini-batch and full-batch.

- Mini-batch algorithm, which applies our offline tri-clustering algorithm to each snapshot of new data matrices. *(high scalability but may sacrifice the quality)*

  Full-batch algorithm, which applies the offline tri-clustering algorithm to the entire dataset whenever new data arrives at each timestamp *(high quality but very time-consuming)*
Comparison (Online)

- Set $w = 2, \quad \beta = 0.8$

Figure 9: Clustering accuracy when varying $\alpha$ and $\tau$ on Proposition 30 data (the figure is best viewed in color)

$\alpha = \tau = 0.9$
Comparison (Online)

Figure 10: Clustering accuracy when varying $\gamma$ for Proposition 30 data
Comparison (Online)

![Comparison Diagram](image)

**Figure 11:** Online performance results for Proposition 30 data (the figure is best viewed in color)

![Comparison Diagram](image)

**Figure 12:** Online performance results for Proposition 37 data (the figure is best viewed in color)
Thank you!