Comment-based Multi-View Clustering of Web 2.0 Items

Xiangnan He, Min-Yen Kan, Peichu Xie, Xiao Chen

Presenter: Xiangnan He
Web IR/NLP Group (WING)
National University of Singapore
User Generated Content: A driving force of Web 2.0

Challenges:
- Information overload
- Dynamic, temporally evolving Web
- Rich but noisy UGC

Daily growth of UGC:
- Twitter: 500+ million tweets
- Flickr: 1+ million images
- YouTube: 360,000+ hours of videos
Comment-based Multi-View Clustering

Why clustering?

Clustering benefits:

– Automatically organizing web resources for content providers.
– Diversifying search results in web search.
– Improving text/image/video retrieval.
– Assisting tag generation for web resources.
Comment-based Multi-View Clustering

Why user comments?

• Comments are rich sources of information:
  – Textual comments.
  – Commenting users.
  – Commenting timestamps.

• Example:

  Comments are a suitable data source for the categorization of web sources!

Figure YouTube video comments
Comment-based Multi-View Clustering

Why user comments?

- Comments are rich sources of information:
  - Textual comments.
  - Commenting users.
  - Commenting timestamps.

Example:

Comments are a suitable data source for the categorization of web sources!

Figure: YouTube video comments
Previous work – Comment-based clustering

- Filippova and Hall [1]: YouTube video classification.
  - Showed that although textual comments are quite noisy, they provide a useful and complementary signal for categorization.
- Hsu et al. [2]: Clustering YouTube videos.
  - Focused on de-noising the textual comments to use comments to cluster.
- Li et al. [3]: Blog clustering.
  - Found that incorporating textual comments improves clustering over using just content (i.e., blog title and body).
- Kuzar and Navrat [4]: Blog clustering.
  - Incorporated the identities of commenting users to improve the content-based clustering.

Previous work – Comment-based clustering

- **Filippova and Hall [1]: YouTube video classification.**
  - Showed that although textual comments are quite noisy, they provide a useful and complementary signal for categorization.

- **Hsu et al. [2]: Clustering YouTube videos.**
  - Focused on de-noising the textual comments to use comments to cluster.

- **Li et al. [3]: Blog clustering.**
  - Found that incorporating textual comments improves clustering over using just content (i.e., blog title and body).

- **Kuzar and Navrat [4]: Blog clustering.**
  - Incorporated the identities of commenting users to improve the content-based clustering.

---

Previous work – Comment-based clustering

• Filippova and Hall [1]: YouTube video classification.
  – Showed that although textual comments are quite noisy, they provide a useful and complementary signal for categorization.

• Hsu et al. [2]: Clustering YouTube videos.
  – Focused on de-noising the textual comments to use comments to cluster.

• Li et al. [3]: Blog clustering.
  – Found that incorporating textual comments improves clustering over using just content (i.e., blog title and body).

• Kuzar and Navrat [4]: Blog clustering.
  – Incorporated the identities of commenting users to improve the content-based clustering.

Inspiration from Previous Work

Both textual comments and identity of the commenting users contain useful signals for categorization.

But no comprehensive study of comment-based clustering has been done to date.

We aim to close this gap in this work.
Problem Formulation

How to combine three heterogeneous views for better clustering?
Experimental evidence

Table 1. Clustering accuracy (%) on the Last.fm and Yelp datasets

<table>
<thead>
<tr>
<th>Method</th>
<th>Last.fm</th>
<th></th>
<th></th>
<th>Yelp</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means (single view)</td>
<td>23.5</td>
<td>30.1</td>
<td>34.5</td>
<td>25.2</td>
<td>56.3</td>
<td>25.0</td>
</tr>
<tr>
<td>K-means (combined view)</td>
<td>40.1 (+5.6%)*</td>
<td></td>
<td></td>
<td>58.2 (+1.9%)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. On a single dataset, different views yield differing clustering quality.
2. For different datasets, the utility of views varies.
3. Simply concatenating the feature space only leads to modest improvement.
4. Same trends result when using other clustering algorithms (e.g., NMF)
Clustering: NMF (Non-negative Matrix Factorization)

Adopted from Carmen Vaca et al. (WWW 2014)
Clustering: NMF (Non-negative Matrix Factorization)

Each entry $W_{ik}$ indicates the degree of item $i$ belongs to cluster $k$. 

Adopted from Carmen Vaca et al. (WWW 2014)
Multi-View Clustering (MVC)

- **Hypothesis:**
  - Different views should admit the same (or similar) underlying clustering.

- **How to implement this hypothesis under NMF?**
Existed Solution 1 – Collective NMF (Akata et al. 2011)

In 16th Computer Vision Winter Workshop, 2011.

• **Idea:**
  – Forcing $W$ matrix of different views to be the same.

\[
V_1 \approx W_1 \times H_1 \\
V_2 \approx W_2 \times H_2 \\
V_3 \approx W_3 \times H_3
\]

• **Drawback:**
  – Too strict for real applications
  (theoretically shown to be equal to NMF on the combined view).
Existed Solution 2 – Joint NMF (Liu et al. 2013)


- **Idea:**
  - Regularizing W matrices towards a common consensus.

- **Drawback:**
  - The consensus clustering degrades when incorporating low-quality views.
Proposed Solution – CoNMF (Co-regularized NMF)

• **Idea:**
  – Imposing the similarity constraint on each pair of views (pair-wise co-regularization).

• **Advantage:**
  – Clustering learnt from each two views complement with each.
  – Less sensitive to low-quality views.
CoNMF – Loss Function

Pair-wise co-regularization:

\[
J_1 = \sum_{s=1}^{n_v} \lambda_s \| V^{(s)} - W^{(s)} H^{(s)} \| + \sum_{s,t} \lambda_{st} \| W^{(s)} - W^{(t)} \|
\]

s.t. \( W^{(s)} \geq 0, H^{(s)} \geq 0 \).

NMF part (combination of NMF each individual view)

Co-regularization part (pair-wise similarity constraint)
Pair-wise CoNMF solution

- **Alternating optimization:**
  
  Do iterations until convergence:
  
  - Fixing $W$, optimizing over $H$;
  
  - Fixing $H$, optimizing over $W$;

- **Update rules:**

  \[
  H^{(s)} \leftarrow H^{(s)} \odot \frac{W(s)^T V(s)}{W(s)^T W(s) H(s)} ,
  \]

  \[
  W^{(s)} \leftarrow W^{(s)} \odot \frac{\lambda_s V(s) H(s)^T}{\lambda_s W(s) H(s) H(s)^T} + \sum_{t=1}^{n_v} \frac{\lambda_{st} W(t)}{\lambda_s W(s) H(s) H(s)^T} + \sum_{t=1}^{n_v} \frac{\lambda_{st} W(s)}{\lambda_s W(s) H(s) H(s)^T}
  \]

  - NMF part: equivalent to original NMF solution.
  - New! Co-regularization part: capturing the similarity constraint.
Normalization Problem

Although the update rules guarantee to converge, but:

1. **Comparable problem**: $W$ matrices of different views may not be comparable at the same scale.

2. **Scaling problem** ($c > 1$, resulting to trivialized descent):

   $$H^{(s)} \leftarrow cH^{(s)}, \quad W^{(s)} \leftarrow \frac{1}{c} W^{(s)}$$

CoNMF loss function:

$$J_1 = \sum_{s=1}^{n_v} \lambda_s \|V^{(s)} - W^{(s)} H^{(s)}\| + \sum_{s,t} \lambda_{st} \|W^{(s)} - W^{(t)}\|,$$

s.t. $W^{(s)} \geq 0, H^{(s)} \geq 0$. 
Normalization Problem

Although the update rules guarantee to find local minima, but:

1. Comparable problem: $W$ matrices of different views may not be comparable at the same scale.

2. Scaling problem ($c > 1$, resulting to trivialized descent):

Address these 2 concerns by incorporating normalization into the optimization process:

– Normalizing $W$ and $H$ matrices per iteration prior to update:

$$W^{(s)} \leftarrow W^{(s)} Q^{(s)-1}, \quad H^{(s)} \leftarrow Q^{(s)} H^{(s)}$$

where $Q$ is the diagonal matrix for normalizing $W$ (normalization-independent: any norm-strategy can apply, such as $L_1$, and $L_2$)
Discussion – Alternative solution

• Alternative solution – Integrating normalization as a constraint into the objective function (Liu et al. SDM 2013):
  – Pros: Convergence is guaranteed.
  – Cons:
    1) Complex – optimization solution becomes very difficult.
    2) Dependent – the solution is specific to the normalization strategy (i.e. need to derive update rules for different norm strategies)

• Our solution – Separate optimization and normalization:
  – Pros:
    1) Simple – Standard and elegant optimization solution derived.
    2) Independent - any normalization strategy can apply.
  – Cons: Convergence property is broken.
**K-means based Initialization**

- Due to the non-convexity of NMF objective function, our solution only finds local minima.
- Research on NMF have found proper initialization plays an important role of NMF in clustering application ([Langville et al. KDD 2006](#)).

- **We propose an initialization method based on K-means:**
  - Using cluster membership matrix to initialize $W$;
  - Using cluster centroid matrix to initialize $H$;
  - Smoothing out the 0 entries in the initialized matrices to avoid the shrinkage of search space.
Datasets

1. **Last.fm**: 21 music categories, each category has 200 to 800 items. In total, about 9.7K artists, 455K users and 3M comments.

2. **Yelp**: a subset of the Yelp Challenge Dataset (7 categories out of 22 categories), each category has 100 to 500 items.

Table 2 Dataset Statistics (filtered, # of feature per view)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Item #</th>
<th>Des.</th>
<th>Com.</th>
<th>Usr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last.fm</td>
<td>9,694</td>
<td>14,076</td>
<td>31,172</td>
<td>131,353</td>
</tr>
<tr>
<td>Yelp</td>
<td>2,624</td>
<td>1,779</td>
<td>18,067</td>
<td>17,068</td>
</tr>
</tbody>
</table>
Experiments

**Baseline Methods for Comparison**

**Single-view clustering methods** (running on the combined view):

1. K-means
2. SVD
3. NMF

**Multi-view clustering methods:**

4. **Multi-Multinomial LDA (MMLDA, Remage et al. WSDM 2009):** extending LDA for clustering webpages from content words and Delicious tags.

For each method, 20 test runs with different random initialization were conducted and the average score (**Accuracy** and F1) is reported.
Results I

Preprocessing

- **Question**: Due to the noise in user-generated comments, how to pre-process the views for better clustering?

<table>
<thead>
<tr>
<th>View</th>
<th>Description</th>
<th>Comment words</th>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>0. Random</td>
<td></td>
<td></td>
<td>6.6</td>
</tr>
<tr>
<td>1. Original</td>
<td>11.8 (+5.3%)</td>
<td>9.3 (+3.3%)</td>
<td>8.4 (+2.2%)</td>
</tr>
<tr>
<td>2. Filtered</td>
<td>15.3 (+4.5%)</td>
<td>9.4 (~)</td>
<td>8.6 (~)</td>
</tr>
<tr>
<td>3. L₁</td>
<td>15.2 (~)</td>
<td>19.0 (+9.7%)</td>
<td>7.9 (~)</td>
</tr>
<tr>
<td>4. L₁-whole</td>
<td>14.5 (~)</td>
<td>9.7 (~)</td>
<td>8.5 (~)</td>
</tr>
<tr>
<td>5. L₂</td>
<td>15.9 (~)</td>
<td>26.9 (+17.5%)</td>
<td>34.5 (+25.9%)</td>
</tr>
<tr>
<td>6. L₂ (tf)</td>
<td>16.8 (~)</td>
<td>25.9 (~)</td>
<td>34.7 (~)</td>
</tr>
<tr>
<td>7. L₂ (tf.idf)</td>
<td>23.5 ( +7.6%)</td>
<td>30.1 (+3.2%)</td>
<td>34.5 (~)</td>
</tr>
<tr>
<td>8. Combined</td>
<td></td>
<td></td>
<td><strong>40.1 (+5.6%)</strong></td>
</tr>
</tbody>
</table>

1. Filtering improves performance and efficiency.
2. L₂ is most effective in length normalization for clustering.
3. TF.IDF is most effective for text-based features.
Results II

Performance Comparison

Effectiveness of CoNMF:
- Performs best in both datasets.
Results III

Consensus - Vs. Pairwise Regularization
CoNMF is stable across a wide range of parameters.
Due to the normalization, we suggest that all regularization parameters are set to 1 when no prior knowledge informs their setting.
**Discussion I**

**Users view utility**

- **Question:** Which users are more useful for clustering?

![Graphs showing accuracy and running time for Last.fm and Yelp](image)

**Figure 5: Accuracy and running time of NMF on the users view**

- **Conclusion:**
  1. Active users are more useful for clustering.
  2. Filtering out less active users improves performance & efficiency.
  3. When the filtering is set too aggressively, performance suffers.
## Discussion II

### Comment-based Tag Generation

**Table 5  Leading words of each cluster**  
(drawn from H matrix of the comment words view)

#### Last.fm

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Top words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambient</td>
<td>ambient, beauti, relax, wonder, nice, music</td>
</tr>
<tr>
<td>Blues</td>
<td>blue, guitar, delta, guitarist, piedmont, electr</td>
</tr>
<tr>
<td>Classical</td>
<td>compos, piano, concerto, symphoni, violin</td>
</tr>
<tr>
<td>Country</td>
<td>countri, tommi, steel, canyon, voic, singer</td>
</tr>
<tr>
<td>Hip hop</td>
<td>dope, hop, hip, rap, rapper, beat, flow</td>
</tr>
<tr>
<td>Jazz</td>
<td>jazz, smooth, sax, funk, soul, player</td>
</tr>
<tr>
<td>Pop punk</td>
<td>punk, pop, band, valencia, brand, untag, hi</td>
</tr>
</tbody>
</table>

#### Yelp

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Top words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active life</td>
<td>class, gym, instructor, workout, studio, yoga</td>
</tr>
<tr>
<td>Arts &amp; Enter.</td>
<td>golf, play, cours, park, trail, hole, theater, view</td>
</tr>
<tr>
<td>Health &amp; Med.</td>
<td>dentist, dental, offic, doctor, teeth, appoint</td>
</tr>
<tr>
<td>Home services</td>
<td>apart, compani, unit, instal, rent, maintain</td>
</tr>
<tr>
<td>Local services</td>
<td>store, cleaner, cloth, dri, shirt, custom, alter</td>
</tr>
<tr>
<td>Nightlife</td>
<td>bar, drink, food, menu, beer, tabl, bartend</td>
</tr>
<tr>
<td>Pets</td>
<td>vet, dog, pet, cat, anim, groom, puppi, clinic</td>
</tr>
</tbody>
</table>
Conclusion and Future Work

• Major contribution:
  – Systematically studied how to best utilize user comments for clustering Web 2.0 items.
    ✓ Both textual comments are commenting users are useful.
    ✓ Preprocessing is key for controlling noise.
  – Formulated the problem as a multi-view clustering problem and proposed pair-wise CoNMF:
    ✓ Pair-wise co-regularization is more effective and robust to noisy views.

• Future work:
  – Can commenting timestamps aid clustering?
Thanks!

QA?
Previous work – Multi-View Clustering (MVC)

• Three ways to combine multiple views for clustering
  
  – Early Integration:
    • First integrated into a unified view, then input to a standard clustering algorithm.
  
  – Late Integration:
    • Each view is clustered individually, then the results are merged to reach a consensus.
  
  – Intermediate Integration
Previous work – Multi-View Clustering (MVC)

• Three ways to combine multiple views for clustering
  – Early Integration:
  – Late Integration:
  – Intermediate Integration:
    • Views are fused during the clustering process.
    • Many classical clustering algorithms have extensions to support such multi-view clustering (MVC)
      *e.g.* $K$-means, Spectral Clustering, LDA

➢ We propose a method to extend NMF (Non-negative Matrix Factorization) for multi-view clustering
Convergence after normalization

• **Without normalization:**
  – In each iteration, the update rules decrease objective function $J_1$.
  – Naturally converge, but may sink into non-meaningful corner cases.

• **With normalization:**
  – In each iteration, $J_1$ is changed before update rules.
  – The update rules decrease $J_1$ with the normalized $W$ and $H$ (*normalized descent*).
  – Not naturally converge (fluctuate in later iterations), but the normalized descent is more meaningful than purely decreasing $J_1$ without normalization.