Comment-based Multi-View Clustering of Web 2.0 Items*

Xiangnan He[1] Min-Yen Kan[1] Peichu Xie[2] Xiao Chen[3]
[1]School of Computing, National University of Singapore
[2]Department of Mathematics, National University of Singapore
[3]Institute of Computing Technology, Chinese Academy of Sciences
{xiangnan, kanmy}@comp.nus.edu.sg xie@nus.edu.sg chenxiao3310@ict.ac.cn

ABSTRACT

Clustering Web 2.0 items (*i.e.*, web resources like videos, images) into semantic groups benefits many applications, such as organizing items, generating meaningful tags and improving web search. In this paper, we systematically investigate how user-generated comments can be used to improve the clustering of Web 2.0 items.

In our preliminary study of Last.fm, we find that the two data sources extracted from user comments – the textual comments and the commenting users – provide complementary evidence to the items' intrinsic features. These sources have varying levels of quality, but we importantly we find that incorporating all three sources improves clustering. To accommodate such quality imbalance, we invoke multi-view clustering, in which each data source represents a view, aiming to best leverage the utility of different views.

To combine multiple views under a principled framework, we propose CoNMF (Co-regularized Non-negative Matrix Factorization), which extends NMF for multi-view clustering by jointly factorizing the multiple matrices through co-regularization. Under our CoNMF framework, we devise two paradigms – pair-wise CoNMF and cluster-wise CoNMF – and propose iterative algorithms for their joint factorization. Experimental results on Last.fm and Yelp datasets demonstrate the effectiveness of our solution. In Last.fm, CoNMF betters k-means with a statistically significant F_1 increase of 14%, while achieving comparable performance with the state-of-the-art multi-view clustering method CoSC [24]. On a Yelp dataset, CoNMF outperforms the best baseline CoSC with a statistically significant performance gain of 7%.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - *Clustering*

Keywords

Comment-based clustering, Multi-view clustering, Co-regularized NMF, CoNMF

*This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

1. INTRODUCTION

With the advent of Web 2.0, the Web has experienced an explosion of user-generated resources. It is reported that there are over 1 million images[1] uploaded to Flickr, and $360,000$ hours[2] of videos uploaded to YouTube per day. To index, retrieve, manage and organize such a large number of web resources accurately and automatically is a major challenge.

Clustering has been an effective method to address this information overload, helping in several different contexts: in automatically organizing web resources for content providers, and in diversifying search results in web document ranking [8]. It has improved retrieval effectiveness for text [41], images [22] and videos [17]. Improved clustering of web resources also helps to automatically generate more meaningful tags [27].

In the context of Web 2.0 and user generated content, how can we cluster such items more effectively? One key observation is the ubiquitous feature of user comments: most Web 2.0 sites enable users to post comments to express their opinions. User comments are a rich source of information, containing not only textual content, but also the commenter's username. Comments' textual content often describes the items in ways complementary to the item metadata, while users themselves are typically interested in a limited range of items matching their interests. As such, user comments are well-suited as an auxiliary data source for tasks. In this paper, we explore the central theme of how to best process user comments and employ them to cluster Web 2.0 items. We believe this research is timely, as recent work [14, 20] have shown that comments do contain useful information in discriminating the categories of items.

As items themselves yield intrinsic features – such as textual description for videos, and pixels for images – how to integrate the two extrinsic data sources derived from comments (here, the textual comments and the commenting users) is an important consideration. A solution might simply build a unified feature space comprising of the features from all three data sources, such that any standard clustering algorithm can then be applied. However, as the three data sources are generated heterogeneously and may vary drastically in clustering quality, a simple combination method may not achieve optimal performance. As such, the key challenge in comment-based clustering is how to meaningfully combine the evidence for clustering. This challenge can be addressed by *multi-view clustering*, where each data source represents a view of possibly different utility.

In this work, we propose extending the NMF (Non-negative Matrix Factorization) for multi-view clustering. NMF [28] factorizes the data matrix in an easily interpretable way and has shown su-

[1]http://www.flickr.com/photos/franckmichel/6855169886
[2]http://www.youtube.com/yt/press/statistics.html

perior performance in document clustering [40]. While substantial research has been conducted on NMF, studies where NMF is used for multi-view clustering are limited. To address this gap, we propose a CoNMF (Co-regularized NMF) framework and offer two instantiations – pair-wise CoNMF and cluster-wise CoNMF. We further derive iterative algorithms for their joint factorization, and apply the factorization results to multi-view clustering.

The main contributions of this paper are in:

- Systematically investigating how to best utilize comments in clustering Web 2.0 items, and formalizing comment-based clustering as a multi-view clustering problem;
- Proposing the CoNMF framework, and two instantiations (pair-wise CoNMF and cluster-wise CoNMF) that extend NMF for multiple views; and
- Applying CoNMF to two real-world datasets, Last.fm and Yelp, and demonstrating the effectiveness of these solutions for comment-based clustering.

The remainder of the paper is organized as follows. After reviewing related work in Section 2, we formalize our research problem and study the problem in a preliminary study on Last.fm in Section 3. In Section 4, we first introduce NMF before proceeding to detail our proposed CoNMF. In Section 5, we evaluate our proposed methods, and discuss some specific topics of comment-based clustering in Section 6. The paper is concluded in Section 7.

2. RELATED WORK

We first review the literature on the general problem of comment-based clustering. We then review work on multi-view clustering, which represents a collection of methods of which our specific proposal of CoNMF is an instance.

2.1 Comment-based Clustering

Comments have been shown to contain useful signals for categorizing and clustering the commented items. Filippova and Hall [14] examined YouTube video categorization. They find that although comments are quite noisy, they do provide useful, complementary and indispensable information for video classification, while the intrinsic features of video title, description and tags are not always indicative of the most relevant category. In a different domain, Li *et al.* [29] cluster blogs, showing that incorporating evidence from the textual content of a blog's comments improves over using the content (*i.e.*, title and body) of the blog alone. Later on, Hsu *et al.* [20] addresses the text of comments, proposing a more comprehensive processing pipeline to de-noise comments. They employ both term normalization and key term extraction before clustering. In [21], Hu *et al.* shows that comments help the summarization of web blogs. While these works are both seminal in showing the efficacy of comments, they only examine the textual content of comments, and ignore the identity of the contributing users, which is a valuable data source for clustering.

To the best of our knowledge, only Kuzar and Navrat's work [25] on Slovak blog clustering has used the identity of the commenting users. They find that users typically comment on similar blogs, and that such implicit relations produce clusterings that differ from content-based clustering. Crucially they show that a combination of both content- and comment-based analyses yields better overall clustering. However, their combination method is heuristic: they first cluster blogs using only blog content. They then identify the decile of blogs with lowest clustering confidence, and refine their clustering based on the commentator-based clustering.

From the above work, we have strong evidence that comments are useful in clustering Web items. However, previous work has yet to comprehensively utilize all parts of the user comments, focusing primarily on the intrinsic content. To the best of our knowledge, no work has yet to provide a comprehensive study of comment-based clustering, nor provided an effective solution to combine the commenting users' identity, textual content from comments, and item-intrinsic features for clustering.

2.2 Multi-View Clustering

Work on multi-view clustering can be grouped into three categories – early, intermediate and late integration – based on when the information from the single views are integrated for clustering.

Early Integration. In these approaches, multiple views are first integrated into a unified view, and then input to any standard clustering algorithm. Representative work include [4, 9], which project the multi-view data into a low-dimensional subspace through Canonical Correlation Analysis (CCA). K-means or spectral clustering is then applied to the projected subspace.

Late Integration. In these approaches, each view is clustered individually, and then the results are merged to reach a consensus. Bo *et al.* [33] assume that the optimal clustering should be close to the clustering of all views as much as possible. Bruno *et al.* [7] treat the optimal clustering as hidden factors to generate the clustering of the different views, and then adopt PLSA [18] to solve the problem. Greene *et al.* [16] first concatenate the cluster membership of different views to a unified matrix, and then perform NMF on the unified matrix to obtain the final clustering.

Intermediate Integration. In these approaches, multiple views are fused during the clustering process. Kumar *et al.* [24] propose a co-regularization framework to extend spectral clustering for multi-view clustering. Wang *et al.* [38] propose a mutual reinforcement clustering approach for multi-view interrelated data objects. Their basic idea is to iteratively propagate the clustering results of one view to all its related views. Ramage *et al.* [36] propose Multi-Multinomial LDA, which extends LDA [5] by assuming the latent factors of each single view are generated by a shared distribution. They show superior performance over k-means on clustering webpages from content words and social tags.

Our proposal directly extends NMF for multi-view clustering, and is an instance of intermediate integration. It is most similar in spirit to [1, 32]. Akata and Thurau [1] propose to jointly factorize multiple data matrices (views) through a shared coefficient matrix (the W matrix in Section 4.1). This is a hard constraint which may be too strict in some scenarios. Additionally, their method is provably equivalent to early integration, where one first concatenates all views into a unified matrix, and subsequently applies NMF. Recently, Liu *et al.* [32] propose MultiNMF, which regularizes the coefficient matrices learned from different views towards a common consensus for clustering. In their work, a key challenge to address is how to make the coefficient matrix of different views comparable. They employ the L_1 norm on the whole data matrix, and then enforce the same L_1 norm constraint on the coefficient matrix during factorization. We find two weaknesses of their solution in practice. First, when the length of vectors varies greatly across views, the resulting proposed L_1 norm on the whole matrix biased towards longer vectors[3]. However, their solution integrates the normalization constraint into the optimization framework, making their technique specific to L_1 norm and difficult to extend to other normalization strategies. Second, when the clustering quality of the component views varies greatly, the learned consensus can underperform a single good view, as the poor quality views negatively affect the consensus. Though one can manually tune weights

[3]Vector length to denote the number of features derived from an item. Section 3.3 and 5.4 demonstrates the impact of normalization on clustering.

to decrease the effect of noisy views, this parameter tuning process of unsupervised learning is non-trivial.

We address both issues of MultiNMF in our method. We co-regularize on each pair of views, which is more robust to the presence of noisy views. This addresses the second issue. For the first issue, we embed the normalization into the optimization process, which enables us to adopt any normalization strategy on the coefficient matrices, effectively offsetting the influence of vector length in multi-view clustering.

3. PRELIMINARIES

Before describing CoNMF, we discuss some necessary preliminaries. We first give a formal problem statement for comment-based clustering, and then introduce the evaluation criteria. We further conduct an initial study on Last.fm that motivates our approach and illustrates the challenges.

3.1 Problem Statement

We investigate how comment data is best used to assist clustering items. We note two separate data sources that can be extracted from comments[4]: the textual content of the comments and the identities of the commenting users. Items also additionally have intrinsic features that can be distilled from the items themselves. Formally, the comment-based clustering problem is then:

Input: A set of items numbered $1, .., m$. Each item consists of three views: a set of words extracted from the textual content of comments, a set of commenting usernames, and intrinsic features derived from themselves. A target number of clusters K.

Output: A mapping from each item to a particular cluster $k \in 1, ..., K$.

Our problem formulation results in a *flat* (non-hierarchical) and *hard* (single-assignment) clustering problem. For soft clustering algorithms, such as LDA and NMF, we take the most likely cluster in the soft assignment to yield a hard assignment. We also note that one can cluster the items based solely on the comments, which can be cast as a two-view clustering problem, a simpler version of our three-view problem. We consider three-view clustering to explore how to best cluster Web 2.0 items with the help of user comments.

3.2 Clustering Evaluation Metrics

Measures for evaluating clustering can be split into *intrinsic* and *extrinsic* criteria. Internally, good clusterings should result in high intra-cluster similarity and low inter-cluster similarity. However, a good score on an intrinsic criterion does not necessarily mean good task effectiveness [34]. For this reason, we adopt extrinsic criteria, which measure how well the clustering matches ground truth (GT). The GT is ideally produced by human judges and with good credibility. In this paper, we evaluate with the extrinsic metrics of clustering accuracy [40] and F_1 [34].

Accuracy measures the percentage of items that are assigned to their correct categories, which is intuitive and one of the easiest means to access clustering quality. The best mapping of clusters to GT labels can be found by the Kuhn-Munkres algorithm [23].

Clustering F_1 is similar to classification F_1, where the only difference is that precision and recall are computed over pairs of items; *e.g.,* a true positive means that a pair of items attributed to the same GT label are correctly assigned to the same cluster. We select F_1 because it measures the quality of putting similar items together while keeping dissimilar items apart, and is well-understood in the information retrieval community. We also employed other metrics – including normalized mutual information, purity and adjusted random index – but as the results are consistent across metrics, we present only accuracy and F_1.

[4]Comment timestamps can also be leveraged, but we leave this extension for future work.

3.3 Preliminary Study

We execute an initial study with data drawn from Last.fm, a music listening and sharing site. We choose Last.fm mainly based on the availability of ground truth, as each item (artist) is tagged with category labels (music genre). Other Web 2.0 sites, such as YouTube, may be a better choice as the items are uploaded by users. However, in these websites the ground truth (categorization of items) may not be of high quality [14, 20], providing an inaccurate evaluation of clustering. We find that the categories of Last.fm artists do accurately reflect their music genre, and thus choose this source for our study. We describe the Last.fm dataset in more comprehensive detail later in Section 5.1, as we use it again in our formal experimentation later.

We utilize the k-means clustering algorithm [35] for our study. K-means is a widely used, intuitive and efficient clustering algorithm based on the vector space model (VSM).

We want to answer the following questions with our study:

Q1. How do the three views differ in their ability to discriminate different categories of items? Do the views based on user comments help?

Q2. How should we preprocess comments to reduce noise and improve clustering efficiency?

Q3. In the VSM, how should each vector be normalized? How should the individual features for each view be weighted?

Q4. How should we combine the three views optimally? Will the resultant combined view yield better clustering?

We run k-means 20 times with random initialization and report the average performance in Table 1 when run with different settings described next. The column names "Des.", "Com." and "Usr." represent the item-intrinsic description view, and the two comment-based views (comment words view and users view), respectively. In answering the above questions, we work our way from the basic k-means to answering the issues of noise filtering, normalization, term weighting and view combination, to yield a worthy baseline for comparison.

Basic Feature Space (Row 1). To get a base result, we first build a plain VSM for each view: each item is represented as a row vector. The raw counts of the words or usernames are used as the vector elements. Then, we run k-means on each view's feature space, yielding the performance reported in Row 1. The clustering quality is poor, bettering random assignment (accuracy / F_1 of about 6.6% / 5.0%) by a small margin.

Filtering Noisy Features (Row 2). As our textual features are known to be noisy, and the feature space is large, we consider how to filter noise to improve performance. For the two text-based views (the comment words and description views), we first retain only English words, then remove common stop words and conflate the words to stemmed form, using the NLTK toolkit [3]. For the users view, we retain users who had commented on more than 2 items, as users that only comment on few items may not be strong signals for clustering. Table 2 shows the dimensionality of the original and reduced feature spaces, where we see a drastic reduction, which aids clustering efficiency. This filtered space's yields improved performance on the description view, while perfomance on the users and comment words views are unchanged. As such, we take the filtered features as the basis for the remainder of this initial study.

Table 1: K-means performance with different settings.

Metric	Accuracy (%)			F_1 (%)		
View	Des.	Com.	Usr.	Des.	Com.	Usr.
1. Basic	11.8	9.3	8.4	7.5	10.1	9.8
2. Filtered	15.3	9.4	8.6	10.9	10.3	9.8
3. L_1	15.2	19.0	7.9	11.0	13.9	9.9
4. L_1-whole	14.5	9.7	8.5	10.8	10.4	9.8
5. L_2 (count)	15.9	26.9	34.5	10.7	17.6	15.2
6. L_2 (tf)	16.8	25.9	34.7	10.6	17.1	15.3
7. L_2 (tf×idf)	23.5	30.1	34.5	14.5	16.8	14.7
8. Combined	**40.1**			**24.2**		

Table 2: **Dimensionality of each view, for the original and reduced feature space.**

View	Des.	Com.	Usr.
Original	$99,405$	$2,244,330$	$455,457$
Reduced	$14,076(-85\%)$	$31,172(-98\%)$	$131,353(-71\%)$

Normalization (Rows 3–5). As normalization influences clustering performance, we assess the impact of different normalization strategies. Item-based L_2 norm, where each item vector is scaled to a unit length, is a widely used scheme for k-means, resulting in Spherical k-means [11]. The item-based L_1 norm yields a unit sum for each vector, which has a probabilistic explanation where feature values represent its probability of occurring in the item, is also often used. In [32], the authors propose using L_1 norm on the whole data matrix (which we denote as L_1-whole), meaning that each entry in the matrix is divided by the sum of all entries. This results in the elements in the entire data matrix summing to unity, which has the probabilistic explanation where each entry denotes the joint probability of the feature and item.

Rows 3–5 show the results of applying these three normalization strategies. While the results for the description view remain largely unchanged, the comment words and users view are improved, with the L_2 norm outperforming both L_1 and L_1-whole significantly. For the description view, we find that the item's description is contributed by the editors of Last.fm, and is of a controlled length. As such, the vector length for different items does not vary much and normalization strategy does not influence the clustering. While for the two comment-based views, the vector length depends on the number of comments on the item, which varies greatly. As shown in Figure 1, although most items ($\sim 95\%$) receive less than 512 comments, these items are almost evenly distributed in different intervals. In such a case, normalizing by L_1-whole will still present a bias towards frequently commented items, while item-based L_2 norm is more effective than L_1 in offsetting the influence of vector length for clustering. As such, in the following, we use the item-based L_2 norm. In other experiments where we substituted NMF for k-means, we reach the same conclusion.

Term weighting (Rows 5–7). Feature weighting also influences the clustering process. In information retrieval, weighting based on term frequency and inverse document frequency (tf×idf) are common. We follow the standards in [2] to implement three common weighting schemes, whose results are shown in Rows 5–7: raw term count (count), term frequency (tf, log of raw term count) and tf×idf. Note that we first weigh the features, before normalizing the vectors with the L_2 norm.

For the two text-based views (description and comment words view), tf×idf performs significantly better than tf and count, while for the users view, all three weighting schemes perform comparably. In the following, we thus use tf×idf for the two text-based views, while using raw term counts for the users view.

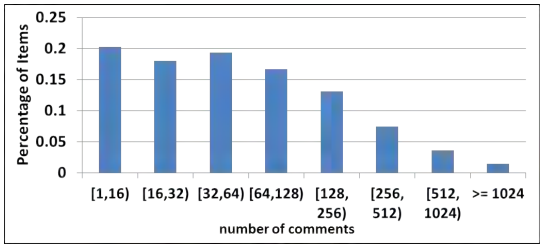

Figure 1: Distribution of items in the Last.fm dataset by number of comments.

Combined view (Row 8). Having benchmarked the clustering performance using the views individually, we assess whether there is benefit in combining the views together using a simple early integration approach. We first normalize each view, and then concatenate all views using the same weight. Formally, let the row vector of an item be $\mathbf{v}_d$, $\mathbf{v}_c$ and $\mathbf{v}_u$ for the three views respectively. Then the combined vector is $\mathbf{v} = [\sqrt{\frac{1}{3}}\mathbf{v}_d, \sqrt{\frac{1}{3}}\mathbf{v}_c, \sqrt{\frac{1}{3}}\mathbf{v}_u]$.

Row 8 shows that such a simple integration performs well, significantly outperforms all of the individual views on both metrics (p-value < 0.01). This results indicates that combining the views is advantageous. Further experiments where we tried different linear weightings of the three views did not further improve performance.

Our preliminary study has benchmarked k-means performance on the clustering of Last.fm artists (items) into genres (categories). We saw that with proper filtering, normalization and feature weighting, the individual views can generate useful clusters and start to answer the four questions posed at the beginning of this section. A key outcome of the study is that the users view (*i.e.*, identity of commenting users) is useful, but potentially overlooked in previous research.

Concluding this preliminary study, we see that early integration by combining all three views into a single view yields improved clustering performance, answering the second half of Q4. But as the views differ in nature and in innate clustering quality, we suspect that a more principled method of integration may yield even better results. The remainder of our paper describes our approach to find a convincing framework for answering Q4.

4. CO-REGULARIZED NMF

Our solution in finding a principled method to combine views adopts the non-negative matrix factorization (NMF) technique. After briefly reviewing on NMF in Section 4.1, we propose the general CoNMF framework to combine multiple views for joint factorization, and then introduce two paradigms of the framework – pairwise CoNMF and cluster-wise CoNMF. As an additional contribution, we further devise a novel k-means based method for CoNMF initialization, and derive the time complexity of our proposed method.

4.1 Non-negative Matrix Factorization

NMF is a matrix factorization technique that factorizes the non-negative data matrix into two non-negative matrices [28]. Formally, let $V \in \mathbb{R}_+^{m \times n}$ be the data matrix of non-negative elements. Each row vector $V_{i\cdot}$ denotes an item (m denotes the number of items and n denotes the number of features). The factorization is formulated as $V \approx WH$, where W and H are $m \times K$ and $K \times n$ matrices, respectively. K is the a pre-specified parameter denoting the dimension of reduced space. In clustering applications, K also denotes the number of desired clusters. The goal of factorization is to

Algorithm 1: Co-regularized NMF (CoNMF)

Input: Non-negative matrices $\{V^{(s)}\}$, parameters $\{\lambda_s\}$, parameters $\{\lambda_{st}\}$ and number of clusters K;
Output: Coefficient matrices $\{W^{(s)}\}$ and basis matrices $\{H^{(s)}\}$;

```
Normalize each view V^(s) such that ||V_i.^(s)|| = 1;
Initialize matrices {W^(s)} and {H^(s)} (Section 4.5);
while Objective function does not converge and
      Number of iterations ≤ Threshold do
    for each s from 1 to n_v do
        Normalize W^(s) and H^(s) using Eq. (12) (Section 4.3.2);
        Update W^(s) and H^(s) using either
            Eq. (10) (Pair-wise CoNMF; cf Section 4.3) or
            Eq. (14) (Cluster-wise CoNMF; cf Section 4.4);
    end
end
return {W^(s)} and {H^(s)}
```

minimize:

$$O = ||V - WH||, \quad s.t. \quad W \geq 0, H \geq 0, \tag{1}$$

where $||\cdot||$ denotes the squared sum of all elements in the matrix. W is termed the *coefficient matrix* and H the *basis matrix*.

It is known that the objective function is not convex in W and H. As such, it is infeasible to find the global minima. In [37], Lee and Seung propose a solution to find a local minima through alternating optimization. The iterative update rules are as follows:

$$H \leftarrow H \odot \frac{W^T V}{W^T W H}, \quad W \leftarrow W \odot \frac{V H^T}{W H H^T}, \tag{2}$$

where $\odot$ and the division symbol in this matrix context denote element-wise multiplication and division[5].

The non-negative property of NMF makes the reduced space easy to interpret, in contrast to other matrix factorizations that do not share this property (*e.g.*, PCA and SVD). Specifically, each element W_{ik} of matrix W indicates the degree of association of item i with cluster k. As such, one just need to take the largest value of row vector $W_{i\cdot}$ as the (hard) cluster assignment of item i. NMF has shown good performance and much work has been done in both applying NMF to different problem areas as well as on studying NMF itself [39]. Aside from the original use of NMF for learning parts of images [28], NMF has shown superior performance in document clustering [40] and website recommendation [30]. Some theoretical studies [13, 15] have shown the equivalence between NMF with other clustering algorithms, including K-means, Spectral Clustering and PLSA, with additional constraints.

4.2 CoNMF Framework

The hypothesis behind multi-view clustering is that different views should admit the same underlying clustering of the data. Formally, given n_v views denoting as $\{V^{(1)}, ..., V^{(n_v)}\}$, each view is factorized as $V^{(s)} \approx W^{(s)} H^{(s)}$, where $W^{(s)}$ are with same dimension $m \times K$ for all views, while $H^{(s)}$ are of dimension $K \times n^{(s)}$, differing per view.

In our CoNMF approach (overview in Algorithm 1), we implement this constraint by coupling the factorization of the views through co-regularization. Generally speaking, the objective function of CoNMF is formulated as:

$$J = \sum_{s=1}^{n_v} \lambda_s ||V^{(s)} - W^{(s)} H^{(s)}|| + R, s.t. W^{(s)} \geq 0, H^{(s)} \geq 0, \tag{3}$$

where λ_s are the parameters to combine the factorization of different views and R is the co-regularization function that enforces similarity constraints on multiple views. CoNMF is a general framework as different regularization schemes and similarity measures can be used to implement the co-regularization function R.

4.3 Pair-wise CoNMF

To implement the hypothesis of multi-view clustering, an intuitive method is to regularize the coefficient matrices of the different views towards a common consensus, which is then used for clustering. This is the cornerstone of MultiNMF [32] (consensus-based co-regularization). However, a key weakness of this approach is that it fares well only when views are largely homogeneous and of roughly the same quality. In real world applications, different views may be generated heterogeneously and may vary drastically in quality. This is the case that we observe in our comment-based clustering settings (*cf.* Table 4 of Section 5.3). In the MultiNMF approach, the model's constraints enforce a rigid common consensus that forces views with higher clustering utility to be degraded by ones with lower utility, which may lead to poorer performance (*cf.* Table 6 of Section 5.4).

Pair-wise CoNMF relaxes MultiNMF's constraints, instead of imposing similarity constraints on each pair of views. Through the pair-wise co-regularization, we expect that the coefficient matrices learned from two views can complement with each other during the factorization process. It should thus yield a better latent space and be more effective for clustering.

Intuitively, the co-regularization function of pair-wise CoNMF is defined as follows:

$$R_1 = \sum_{s=1}^{n_v} \sum_{t=1}^{n_v} \lambda_{st} ||W^{(s)} - W^{(t)}|| = \sum_{s,t} \lambda_{st} ||W^{(s)} - W^{(t)}||, \tag{4}$$

where λ_{st} is the parameter to denote the weight of the similarity constraint on $W^{(s)}$ and $W^{(t)}$. Substituting R in Eq. (3) with R_1, we obtain the objective function:

$$J_1 = \sum_{s=1}^{n_v} \lambda_s ||V^{(s)} - W^{(s)} H^{(s)}|| + \sum_{s,t} \lambda_{st} ||W^{(s)} - W^{(t)}||, \quad s.t. \quad W^{(s)} \geq 0, H^{(s)} \geq 0. \tag{5}$$

We then minimize the objective function to get the solution.

4.3.1 Optimization

Similar to the known solution for NMF, we can adopt alternating optimization to minimize the objective function. The optimization works as follows: (1) fix the value of $W^{(s)}$ while minimizing J_1 over $H^{(s)}$; then (2) fix the value of $H^{(s)}$ while minimizing J_1 over $W^{(s)}$. We iteratively execute these two steps until convergence, or until a set number of iterations is exceeded.

The objective function J_1 can be re-written as:

$$\begin{aligned} J_1 = & \sum_{s=1}^{n_v} \lambda_s Tr(V^{(s)T} V^{(s)} - 2V^{(s)T} W^{(s)} H^{(s)} \\ & + H^{(s)T} W^{(s)T} W^{(s)} H^{(s)}) \\ & + \sum_{s,t} \lambda_{st} Tr(W^{(s)T} W^{(s)} - 2W^{(s)T} W^{(t)} + W^{(t)T} W^{(t)}), \end{aligned} \tag{6}$$

[5]For example, $(A \odot B)_{ij} = A_{ij} B_{ij}$. Same for element-wise division. We adopt this expression in the following sections.

where $Tr(\cdot)$ denotes the trace function. Here, $||A|| = Tr(A^T A)$ and $Tr(AB) = Tr(BA)$ are used in the derivation. To enforce the non-negativity constraints, we need to incorporate Lagrange multipliers. Let $\alpha^{(s)}$ and $\beta^{(s)}$ be the Lagrange matrices for constraint $W^{(s)} \geq 0$ and $H^{(s)} \geq 0$, respectively. The Lagrange L_1 is:

$$L_1 = J_1 + \sum_{s=1}^{n_v} Tr(\alpha^{(s)} W^{(s)T}) + Tr(\beta^{(s)} H^{(s)T}). \quad (7)$$

Then, the derivatives of L_1 with respect to $W^{(s)}$ and $H^{(s)}$ are:

$$\begin{aligned}\frac{\partial L_1}{\partial W^{(s)}} =& \lambda_s(-2V^{(s)}H^{(s)T} + 2W^{(s)}H^{(s)}H^{(s)T}) \\ &+ \sum_{t=1}^{n_v} \lambda_{st}(2W^{(s)} - 2W^{(t)}) + \alpha^{(s)}, \quad (8)\\ \frac{\partial L_1}{\partial H^{(s)}} =& \lambda_s(-2W^{(s)T}V^{(s)} + 2W^{(s)T}W^{(s)}H^{(s)}) + \beta^{(s)}.\end{aligned}$$

Using the Karush-Kuhn-Tucker (KKT) conditions that $\alpha_{ij}^{(s)} W_{ij}^{(s)} = 0$ and $\beta_{ij}^{(s)} H_{ij}^{(s)} = 0$, we have:

$$\begin{aligned}\frac{\partial L_1}{\partial W^{(s)}} \odot W^{(s)} =& 0, \\ \frac{\partial L_1}{\partial H^{(s)}} \odot H^{(s)} =& 0.\end{aligned} \quad (9)$$

Solving the above equations, we derive the following update rules:

$$\begin{aligned}H^{(s)} &\leftarrow H^{(s)} \odot \frac{W^{(s)T}V^{(s)}}{W^{(s)T}W^{(s)}H^{(s)}}, \\ W^{(s)} &\leftarrow W^{(s)} \odot \frac{\lambda_s V^{(s)}H^{(s)T} + \sum_{t=1}^{n_v} \lambda_{st} W^{(t)}}{\lambda_s W^{(s)}H^{(s)}H^{(s)T} + \sum_{t=1}^{n_v} \lambda_{st} W^{(s)}}.\end{aligned} \quad (10)$$

These update rules form the solution for the pair-wise CoNMF algorithm's iterative execution. It is easy to see that $W^{(s)}$ and $H^{(s)}$ are non-negative after each update. Moreover, it is provable that the objective function J_1 is non-increasing under the above iterative updating rules, and the convergence is guaranteed. The proof can be shown by constructing the auxiliary function similar to [37][6].

4.3.2 Normalization

While the above provides a sound solution for the optimization, in practice we find that inserting a normalization step is important. The above solution is guaranteed to minimize the objective function with local minima, but we notice that this solution does not always lead to meaningful results. There are two possible reasons for this: (1) the W matrices of the different views might not be comparable at the same scale; (2) there is a case that the value of objective function is always decreased by scaling, but which does not yield meaningful progress towards a solution. To see the case, let us consider a solution $W^{(s)}$ and $H^{(s)}$. In the next iteration, the value of J_1 can be decreased by the update:

$$H^{(s)} \leftarrow cH^{(s)}, \quad W^{(s)} \leftarrow \frac{1}{c}W^{(s)}, \quad (11)$$

where c is a constant larger than 1. Under these update rules, the first term of J_1 in Eq. (5) (the combination of factorization of different views) remains unchanged, while the second term (co-regularization function) is decreased. In this case, J_1 is decreased through just scaling the $W^{(s)}$ and $H^{(s)}$, which is not meaningful.

[6]The proof is provided in the supplementary materials at `http://www.comp.nus.edu.sg/~xiangnan`

We can solve both problems by normalizing the W matrices of the different views to make them comparable with each other, and effectively disallowing scaling. Notice that each column vector of $W^{(s)}$ represents a cluster, whose elements give the strength of association of the items to the cluster. As such, normalizing the column vectors of $W^{(s)}$ makes the cluster assignments of different views comparable. As our preliminary analysis (Section 3.3) has shown that the vector based L_2 norm is more effective in offsetting the influence of vector length for clustering, we adopt the L_2 norm.

Formally, let $Q^{(s)}$ be the diagonal matrix with values $Q_{jj}^{(s)} = \sqrt{\sum_i W_{ij}^{(s)2}}$. Then the normalization strategy works as follows:

$$W^{(s)} \leftarrow W^{(s)}Q^{(s)-1}, \quad H^{(s)} \leftarrow Q^{(s)}H^{(s)}. \quad (12)$$

Note that $H^{(s)}$ is scaled by $Q^{(s)}$ correspondingly. In applying this simultaneous normalization, the value of the first term of Eq. (5) remains unchanged, while the co-regularization function is then forced to become meaningful as the coefficient matrices from different views are comparable.

With this modified procedure, we first normalize the W and H matrices of all views, and then execute the update rules during each iteration. In each iteration, the update rules decrease the value of J_1 with the normalized W and H (we term it *normalized descent*). While the normalization process may change the original value of J_1 before updating, the algorithm might not be naturally converged. However, we argue that this normalized descent is more meaningful than purely decreasing the value of J_1, because it avoids both the comparable problem and scaling problem.

4.4 Cluster-wise CoNMF

Adopting the L_2 normalization admits another possible implementation of CoNMF. As the column vector of the coefficient matrix W represents a cluster, when we adopt the vector-based L_2 norm, each entry of $W^T W$ gives the cosine similarity between two clusters. As such, $W^T W$ can then be interpreted as the pair-wise cluster similarity matrix.

This leads to a natural definition for a cluster-wise paradigm of CoNMF. We define the co-regularization function of cluster-wise CoNMF as follows:

$$R_2 = \sum_{s,t} \lambda_{st} ||W^{(s)T}W^{(s)} - W^{(t)T}W^{(t)}||. \quad (13)$$

Following the same process of optimization as in Section 4.3.1, we obtain the following update rules for cluster-wise CoNMF:

$$\begin{aligned}H^{(s)} &\leftarrow H^{(s)} \odot \frac{W^{(s)T}V^{(s)}}{W^{(s)T}W^{(s)}H^{(s)}}, \\ W^{(s)} &\leftarrow W^{(s)} \odot \frac{\lambda_s V^{(s)}H^{(s)T} + 2\sum_t \lambda_{st} W^{(s)}W^{(t)T}W^{(t)}}{\lambda_s W^{(s)}H^{(s)}H^{(s)T} + 2\sum_t \lambda_{st} W^{(s)}W^{(s)T}W^{(s)}}.\end{aligned} \quad (14)$$

Note that the update rules for $H^{(s)}$ of both CoNMF instantiations are the same, and are equivalent to standard NMF. This is because our proposed CoNMF only makes soft regularization with respect to the W matrices, while the H matrices – which represent the factorization of each individual view – remain unchanged. This desireable property effectively retains the information of each view during the factorization process. We discuss this property in Section 5.4.

4.5 Initialization

As the objective function of NMF is non-convex, the iterations only find locally-optimal solutions. Under standard NMF, W and H are initialized randomly. However, research on NMF have found

that proper initialization plays an important role in the performance of NMF in many applications [6, 26]. It is reported that all NMF algorithms are sensitive to the initialization [26].

With multi-view clustering in mind, we propose a method to initialize CoNMF more effectively based on k-means, which is simple and efficient. Running k-means yields two outputs: the cluster assignment of each item and the centroid of each cluster. We propose to use these outputs to initialize W and H, respectively. We initialize the W matrix uniformly for all views while initializing the H matrix separately for each view. This is because the W matrices will be softly regularized with each other, while the H matrices are updated separately to represent the factorization of each view.

Initialization of W matrices. To initialize W, we first run k-means on the combined view. The clustering assignments can be represented as a $m \times K$ cluster membership matrix M, such that $M_{ik} = 1$ if and only if item i is assigned to cluster k, otherwise $M_{ik} = 0$. As W is the coefficient matrix denoting the cluster membership, M can be used to initialize W. We propagate the $M_{ik} = 1$ entries as-is in $W^{(s)}$, but importantly, set all $M_{ik} = 0$ entries to a random number r in the range $(0, 1)$, instead of 0. This is needed to prevent the search space from becoming too sparse prematurely, as under the multiplicative CoNMF update rules, zero entries lead to a disconnected search space and result in overly localized search. The proposed initialization smooths out the initial search space, dealing with sparsity, while conforming to the same k-means combined view clustering in the first iteration.

Initialization of H matrices. For the initialization of each $H^{(s)}$, we first run k-means on the view s. Let the centroid of a cluster be a vector $\mathbf{c}_k^{(s)}$, then all centroids of the clustering can be represented as a matrix $C^{(s)} = [\mathbf{c}_1^{(s)}, ..., \mathbf{c}_K^{(s)}]^T$. We use $C^{(s)}$ as the initialization of $H^{(s)}$. The reasons are as follows. The factorization of NMF can be written as

$$V_{i\cdot} \approx \sum_{k=1}^{K} W_{ik} H_{k\cdot}, \tag{15}$$

where $V_{i\cdot}$ is the i-th row vector of data matrix V, $H_{k\cdot}$ is the k-th row vector of H. As such, $H_{k\cdot}$ can be seen as the basis vector to resemble the original data. In k-means clustering, each item is assigned to the cluster with nearest centroid. Therefore, the centroids of k-means clustering can also be deemed as the K basis vectors of the original data. As such, using the centroids to initialize H places them in the same space initially, which is more meaningful than random initialization. Similarly, as the update rules of $H^{(s)}$ are multiplication-based and $C^{(s)}$ may be very sparse, which may cause shrinkage of the search space. We add a small constant ϵ to each element of $C^{(s)}$ to avoid the shrinking effect.

4.6 Time Complexity Analysis

We now analyse CoNMF's time complexity, using standard NMF as the basis for big O notation.

CoNMF is essentially an extension of NMF for multiple data matrices. It can be shown that the cost for NMF's update rules in each iteration is $O(nmK)$. As CoNMF's update rule for each $H^{(s)}$ is same with the original NMF, its cost is also $O(nmK)$. For each $W^{(s)}$ of pair-wise CoNMF in Eq. (10), the additional cost in terms of plain NMF is the second term of the numerator and denominator, whose time complexity is $O(n_v mK)$. As such, the time complexity of update rules of pair-wise CoNMF is $O(n_v mK + nmK)$. As n_v denotes the number of views, which is a small constant (in our comment-based clustering, $n_v = 3$) *s.t.* $n_v \ll n$, this yields $O(n_v mK + nmK) \approx O(nmK)$. Similarly, for cluster-wise CoNMF, the time complexity of update rules of each view is $O(n_v mK^2 + nmK) \approx O(nmK)$. Therefore, the time complexity of CoNMF update rules in each iteration is $O(n_v nmK)$, as there are n_v views to update, making CoNMF a linear extension of NMF. We empirically verified this in our experiments, as the actual running time of CoNMF was similar to running plain NMF on the three single views in series.

In real applications, although n may be very large, the data matrix is typically very sparse. As such, the number of actual operations can be far less. In addition, the multiplication-based update rules of our proposed CoNMF solutions further reduce the calculation, especially in later iterations. Distributed computation strategies for NMF with MapReduce [30] can also be used on CoNMF, ensuring that CoNMF can also be applied to large-scale data.

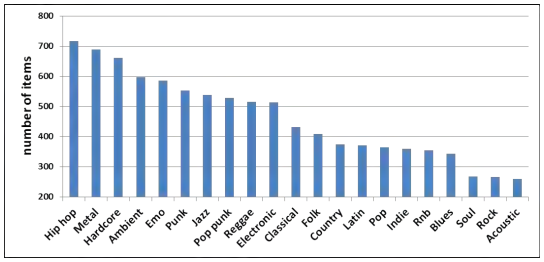

Figure 2: Items per category in our Last.fm dataset.

5. EXPERIMENTS

Our evaluation focuses on evaluating CoNMF for comment-based multi-view clustering; specifically, to quantify the performance gain by utilizing the signal across views. We do this by first benchmarking the performance computed from single views, then contrasting it against the performance on multi-view clustering. We also compare CoNMF against other multi-view clustering techniques.

5.1 Datasets

We experiment with two datasets: Last.fm and Yelp. Table 3 gives summary demographics over the two datasets.

Last.fm. This dataset is the source of our preliminary study described earlier. Last.fm lists 26 music genres. We use 21 of these, which are shown in Figure 2. We exclude "world", "60s", "70s", "80s", "90s", which we feel are less reflective of a particular music style. For each of the 21 genres' music page, we crawl the artists tagged to it. As an artist may be tagged with multiple genres, we retain only artists tagged to a single genre, to facilitate hard clustering evaluation. For each artist, we crawl his or her bio description and user comments. In total, our Last.fm dataset consists of $9,694$ artists, $455,457$ users and $2,993,222$ comments. Figure 2 shows the distribution of items (artists) to genre in our Last.fm dataset.

After the reduction on features described in Section 3.3, we arrive at a reduced set of $14,076$ description features (unique tokens), $31,172$ comment features and $131,153$ unique users. The following experiments are on the reduced dataset.

Yelp. This dataset is a subset of the Yelp Challenge Dataset (YDC)[7], which is from the greater Phoenix, AZ metropolitan, including $11,537$ items (businesses), $229,907$ comments and $43,873$ users. Each item is associated with relevant categories, from a fixed vocabulary provided by Yelp. There are 22 first-level categories. Retaining only items that are unambiguously mapped to only one first-level category, we obtain $9,537$ items. Figure 3 shows the statistics of number of items per category on this dataset. As can be seen, the distribution is very skewed: the top category "restaurants" takes 39.9% items and the top three categories take 64.5%

[7] http://www.yelp.com/dataset_challenge

Table 3: Per-view demographics for our datasets.

Dataset	Item #	Des.	Com.	Usr.
Last.fm	9,694	14,076	31,172	131,353
Yelp	2,624	1,779	18,067	17,068

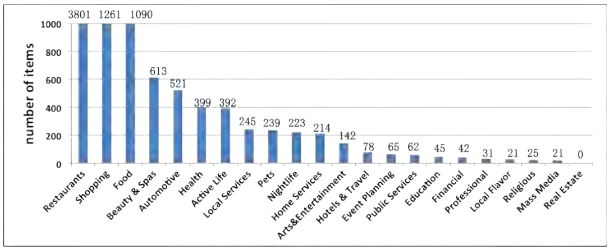

Figure 3: Items per category in our Yelp dataset.

items. Such a skewed distribution influences the clustering evaluation greatly. To balance the number of items per category, one common way is to randomly sample some items for the large categories [32, 24]. However, this makes evaluation unstable and hard to replicate. As such, we further limit our dataset to categories with that have only items in the range of 100 to 500. Our final Yelp dataset consists of 2,624 items from 7 categories: *Health & Medical*, *Active Life*, *Local Services*, *Pets*, *Nightlife*, *Home Services* and *Arts & Entertainment*. This dataset consists of three views as well. The comment words view and users view are extracted the same way as in Last.fm, with the exception that we drop the users view frequency filter, as the dataset is smaller in general. For the item-intrinsic view (description view), we use the businesses' names.

5.2 Baselines

We implement CoNMF on the basis of nimfa [42], a python library for NMF. Aside from the baseline k-means and NMF, we further compare with the following algorithms:

1. SVD. We run SVD on the data matrix, using the objective latent number of dimensions as K, then cluster the reduced space using k-means. This is a typical SVD workflow for clustering [40].

2. MMLDA [36]. Multi-Multinomial LDA is an extension of LDA for clustering webpages from content words and social tags, which can be seen as two views. Latent topics of words and tags are generated from the same multinomial distribution. As it is a two-view clustering algorithm, we merge the two text-based views (description and comment words view) into a single "words" view, then run the algorithm on the words view and users view, to derive the final clustering. We use the EM implementation of [10]. The topic prior is set to be 0.7, as suggested by the authors.

3. CoSC [24]. This is a co-regularization based extension of spectral clustering algorithm, designed specifically for multi-view clustering. We use the default Gaussian kernel to build the affinity matrix and set the regularization parameters to be 0.01, as suggested by the authors.

4. MultiNMF [32]. This is a consensus-based regularization solution for NMF on multi-view clustering. As the authors provide a NMF-based initialization, we use their suggested initialization method, setting the regularization parameters uniformly as 0.01 as suggested. Trying other values, we also find its performance to be consistent. Initially, MultiNMF normalizes the data matrix using L_1-whole, which has been shown to be sensitive to the vector length. For this reason, we further evaluate a solution that attempts to remove the influence of vector length. This solution, which we term, MultiNMF-L_2, first conducts item-based L_2 norm before L_1-whole, and then runs MultiNMF.

Table 4: Single-view clustering results. The best performing algorithm's results are bolded.

Metric	Accuracy (%)			F_1 (%)		
View	Des.	Com.	Usr.	Des.	Com.	Usr.
Last.fm						
k-means	23.5	30.1	34.5	14.5	16.8	14.7
SVD	28.2	27.6	28.0	**24.5**	23.4	24.5
NMF	**29.5**	**39.1**	**43.6**	17.4	**28.0**	**31.6**
Yelp						
k-means	25.2	56.3	**25.0**	26.6	50.2	**26.4**
SVD	23.7	23.8	19.6	22.3	22.8	19.8
NMF	**37.2**	**60.2**	23.6	**27.5**	**57.0**	21.5

Table 5: Multi-view clustering results (mean $\pm$ standard deviation with 95% confidence intervals).

Dataset	Last.fm		Yelp	
Metric	Acc. (%)	F_1 (%)	Acc. (%)	F_1 (%)
k-means	40.1 ± 2.5	24.2 ± 1.9	58.2 ± 7.2	52.2 ± 6.5
SVD	29.7 ± 4.5	24.2 ± 3.1	23.0 ± 1.8	21.5 ± 2.4
NMF	45.5 ± 3.2	35.6 ± 1.9	58.5 ± 6.8	51.8 ± 5.6
MMLDA	35.2 ± 1.6	27.5 ± 1.5	48.1 ± 7.3	47.1 ± 6.8
CoSC	**51.7±2.3***	**38.9±1.7***	60.8 ± 2.7	56.4 ± 3.0
MulNMF	29.9 ± 1.8	21.6 ± 1.3	31.6 ± 2.4	24.2 ± 1.5
MulNMF-L_2	45.5 ± 2.3	31.7 ± 1.6	30.2 ± 2.6	24.8 ± 1.5
CoNMF-P	**51.9±2.5***	**38.8±1.8***	**67.6±4.6***	**63.8±3.7***
CoNMF-C	49.7 ± 2.5	36.2 ± 1.8	**67.3±5.4***	**63.6±4.9***

For fair comparison, we consider all three views as equally important in our comment-based clustering. In the CoNMF settings, the regularization parameters are set to 1 for all views and datasets. We study the parameter settings in Section 5.4.1. As the W matrix of either view can be used for clustering, we report the performance of the best view. For each method, 20 test runs with different random initializations were conducted and the average score is reported. In the following, we report statistical significance (judged at the 5% level by a one-tailed two-sample t-test) where appropriate.

5.3 Single-view Clustering

Running clustering on the single views establishes a baseline for comparison against multi-view clustering. It also allows us to compare the different single view clustering algorithms: k-means, SVD and NMF.

For Last.fm (Table 4, top), NMF achieves the best performance most often. The performance variation across different views is consistent in k-means and NMF: the users view performs best, and the description view performs worst. SVD, in contrast, yields consistent sub-par performance across all views, even when we vary the K for the number of latent dimensions (not shown). As SVD maps the data into orthogonal bases, which may lead to negative values, SVD's clusters are difficult to interpret naturally [40]. Thus, it is inappropriate to judge clustering credibility of the views. The results of SVD on the Yelp dataset also reflect this.

For Yelp (Table 4, bottom), the comment words view performs best, and the users view performs worst. Additionally, the gap between different views' performance are larger than those for Last.fm. We posit that the disparity will challenge standard multi-view clustering algorithms, as the views with poor performance may degrade the clustering of the well-performing views.

5.4 Multi-view Clustering

Table 5 shows the results of multi-view clustering. K-means,

Table 6: Effect of two regularization schemes on the clustering accuracy (%) of each single view.

Dataset	Last.fm			Yelp		
View	Des.	Com.	Usr.	Des.	Com.	Usr.
MulNMF-L_2	43.4	45.0	44.8	29.8	30.9	28.9
CoNMF-P	33.2	42.4	**51.9**	50.2	**67.6**	43.4

SVD and NMF are run on the combined view. CoNMF-P achieves the best performance in all cases, while CoSC and CoNMF-C achieve comparable performance on Last.fm and Yelp, respectively. Although the difference between CoNMF-P and CoNMF-C is less salient for Last.fm, it is consistent and statistically significant.

We also note that the standard deviation in Yelp is generally larger than Last.fm, which we attribute to the larger performance gap in the single view clustering: the performance gap (accuracy / F_1) in terms of k-means between the comment words and users view is 31.3% / 23.8%; in contrast, the largest gap in Last.fm (between users and description views) is 11.0% / 0.2%.

Single view clustering on the combined view leads to mixed results: sometimes better and sometimes worse. SVD does not show significant improvement, k-means improves only for Last.fm, and NMF does better for Last.fm but worse for Yelp. This provides evidence that when views differ in quality, simply combining all views may not lead to improved performance.

Surprisingly, MMLDA underperforms the single view clustering of k-means and NMF. A plausible explanation is that the assumption of shared distribution to generate the latent topics of words view and users view may not hold for comment-based clustering. MMLDA was originally proposed to combine words and tags for webpage clustering. Words and tags are all text-based features, which are used to describe webpages and are still homogeneous. However in comment-based clustering, the users view and the words view are entirely different in nature: the users view reflects the users who are interested in a range of items, while the words view describe items. As such, the shared distribution constraint of MMLDA may be too hard, and a soft constraint may perform better.

MultiNMF does not outperform the single view baselines significantly. We believe both the normalization and regularization strategies of MultiNMF may be responsible. For normalization, MultiNMF proposes to use L_1-whole, which is sensitive to vector length. As can be seen in Last.fm, the original MultiNMF does not perform well, but that applying item-based L_2 norm before L_1-whole works better. In consensus-based regularization, multiple views are regularized towards a common consensus, which may decrease performance when incorporating views with lower quality. The Yelp results provide evidence for this case: NMF on the best (worst) view yields an accuracy of 60.2% (23.6%), and the resultant MultiNMF only achieves 31.6% accuracy. The large performance gap between CoNMF and MultiNMF in Yelp dataset supports our claim that pair-wise co-regularization suffers less from noisy views, and that the joint factorization generates a better latent space for more effective clustering.

To demonstrate the difference of two regularization schemes, we show the clustering accuracy of each single view after regularization in Table 6. After the consensus-based regularization of MultiNMF, each view obtains similar performance and reaches a consensus. However, the information of a view itself is lost due to the consensus constraints. In contrast, CoNMF retains the performance variance across views is similar to the original NMF (Table 4), while improving each view's clustering performance over NMF. It is this ability that leads to the overall improvement of CoNMF over MultiNMF as in Table 5.

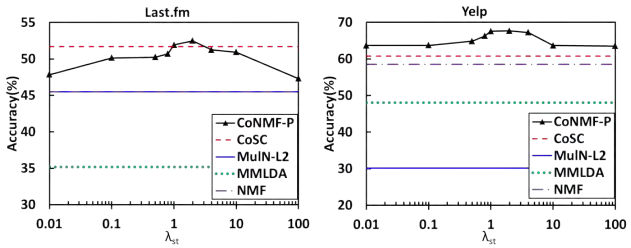

Figure 4: Evaluation on λ_{st} while holding $\lambda_s = 1$ for all views.

Overall, the results demonstrate the effectiveness of CoNMF for comment-based multi-view clustering. By combining all three views in a principled way, CoNMF performs consistently better than clustering in single views as well as in the combined view. In Last.fm, CoNMF achieves a comparable performance with state-of-the-art method CoSC, and outperforms other baselines significantly. In Yelp, CoNMF performs best and achieves about 7% performance gain over the best baseline, CoSC.

5.4.1 CoNMF Parameter Study

There are two sets of regularization parameters in CoNMF: λ_s for each view, and λ_{st} for each pair of views. Relative λ_s values determine each view's importance in factorization; while relative λ_{st} values determine the weight of the pair's similarity constraint in co-regularization. Relative values across λ_s and λ_{st} balance the effect of factorization and co-regularization.

By default, all parameters are set to 1. Figure 4 shows the performance of CoNMF-P when varying λ_{st} while holding $\lambda_s = 1$ for all views. We report only the accuracy of CoNMF-P, as F_1 figures and CoNMF-C are similarly consistent. As can be seen, for both datasets, CoNMF-P is relatively stable across a wide spectrum of settings, performing best when λ_{st} in the 1–2 range. Specifically, for Last.fm across all settings, CoNMF-P betters other baselines besides CoSC (best performance obtained when $\lambda_{st} = 2$, which is 52.5%, but is still in the same significance level with CoSC). In Yelp, over all parameter settings, the performance is significantly better than all baselines. As the three views have different clustering credibility, we also studied whether we can improve the clustering by tuning the weight λ_s of the best view. However, the performance is not improved.

These results indicate that CoNMF is stable across a wide range of parameters. As the coefficient matrices are normalized before the update rules at each iteration, they are already comparable for co-regularization. This suggest that both sets of parameters can be set to 1 when no prior knowledge informs their setting.

6. DISCUSSION

We examine two specific topics worth a more detailed discussion: on the utility of the users view for comment-based clustering, and how clustering could be applied to tag generation (a topic of much current interest).

6.1 Users View Utility

Intuitively, the utility of the users view relies on users commenting on like items, which provides evidence for clustering. The users view is most effective for users who selectively comment only many items in a single category. However, when users comment on either only one item, the value of their comment action (*n.b.*, just the action, and not the content) is zero.

We can filter users by comment frequency to try to favor the former case. We set a comment frequency threshold t, filtering out users who comment less frequently than the threshold from the

Table 7: Sample prominent words drawn from the clusters of the comment words view.

Last.fm		Yelp	
Cluster	**Top words**	**Cluster**	**Top words**
Ambient	ambient, beauti, relax, wonder, nice, music	Active life	class, gym, instructor, workout, studio, yoga
Blues	blue, guitar, delta, guitarist, piedmont, electr	Arts & Enter.	golf, play, cours, park, trail, hole, theater, view
Classical	compos, piano, concerto, symphoni, violin	Health & Med.	dentist, dental, offic, doctor, teeth, appoint
Country	countri, tommi, steel, canyon, voic, singer	Home services	apart, compani, unit, instal, rent, mainten
Hip hop	dope, hop, hip, rap, rapper, beat, flow	Local services	store, cleaner, cloth, dri, shirt, custom, alter
Jazz	jazz, smooth, sax, funk, soul, player	Nightlife	bar, drink, food, menu, beer, tabl, bartend
Pop punk	punk, pop, band, valencia, brand, untag, hi	Pets	vet, dog, pet, cat, anim, groom, puppi, clinic

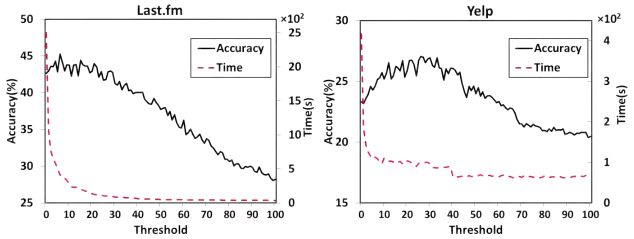

Figure 5: Accuracy and running time of NMF on the users view

original datasets. Figure 5 shows how the performance and running time of NMF vary with threshold t. As CoNMF extends NMF, the performance–time curve for CoNMF is consistent with NMF. We observe that a small amount of filtering is significantly useful in lessening the computational costs for NMF on the users view. As a case in point, when $t = 20$, only 2.7% and 1.4% of the original users remain in the users view of the two datasets. In such cases, the filtered users do not contribute much signal, and may even filter noise, while filtering them out leads to better performance (as seen in the Yelp dataset for $10 \leq t \leq 30$). When filtering is set too aggressively, as expected, we start to lose signal and accuracy drops. As a result, we draw the conclusion that modest filtering helps to boost efficiency while dropping ineffective users.

6.2 Comment-based Tag Generation

In CoNMF, W matrix is the reduced latent space of items, while H matrix serves as the basis matrix for representing a view. As each base (row vector of H) represents a cluster, the leading elements of each base are most representative of the cluster. As the comment words view's elements correspond to comment tokens, we can thus identify representative words in the comments for each cluster. Table 7 shows the words that are mapped to the leading elements in H for the comment words view. For convenience, we automatically map a cluster to a category name by using the Kuhn-Munkres algorithm, shown in the "Cluster" columns. These results show that CoNMF often identifies meaningful words to represent a cluster. We also generated the top words derived from the description view (not shown), finding that the identified words are often complementary to those from comments. Our manual assessment is that the ones derived from the comments are better general descriptors for both datasets. This may be caused by the superior clustering performance of the comment words view has over the description view.

This facility of CoNMF can be utilized in downstream applications, such as tag generation. Approaches might use the top-ranked words as tags directly, or use the values in H as weights into a more sophisticated tag generation algorithm [31]. In related work, Lappas *et al.* [27] has shown that item–aspect distribution learned from social networks can improve tag generation. As the coefficient matrix resulting from CoNMF can be seen as the item–aspect distribution (after normalization via L_1 norm), we believe CoNMF's improved clustering will also lead to improved tag generation.

7. CONCLUSION AND FUTURE WORK

We have systematically investigated how to best utilize user comments for clustering Web 2.0 items, a core task to several information retrieval and web mining applications. In an initial study on Last.fm, we show that the information extracted from user comments – the textual comments and the commenting users – provide complementary information to items' intrinsic features. Combining all three sources of information improves clustering performance over using intrinsic features alone.

Spurred by this result, we formalize this problem as a multi-view clustering problem. We first propose a general framework, CoNMF, as an extension to NMF that combine multiple views for joint factorization. Two paradigms of CoNMF – pair-wise and cluster-wise – are then introduced. Experiments on Yelp and Last.fm datasets show that CoNMF effectively makes use of information from user comments for the clustering task.

In the future, we will study whether including comment timestamps can aid clustering, as user interests may evolve with time. We would like to evaluate the impact of our comment-based clustering on tasks such as web search ranking, recommendation and automatic tag generation. As this work extends the original NMF for multi-view clustering, which naturally assumes that for each view, the number of clusters is the same for the items and features. However, this constraint may be suboptimal, as the items and features of different views have different semantics and may be better described using a different number of clusters per view. Tri-factorization [12] can address this constraint and may improve the performance, which we reserve for future work. Other extensions, which have been shown useful for NMF-based clustering techniques, such as adding orthogonality [12] and sparsity constraints [19], will be explored for CoNMF. Moreover, as our proposed CoNMF is a general approach, having a wider applicability in modeling data with multiple signals, we plan to study its performance in other applications, such as Twitter and Facebook streams.

8. ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for their valuable comments, and wish to acknowledge the additional proofreading and discussions with Jun-Ping Ng, Aobo Wang, Tao Chen, Ming Gao and Jinyang Gao.

9. REFERENCES

[1] Z. Akata, C. Thurau, and C. Bauckhage. Non-negative matrix factorization in multimodality data for segmentation and label prediction. In *16th Computer Vision Winter Workshop*, 2011.

[2] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*. ACM Press New York, 1999.

[3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. O'reilly, 2009.

[4] M. B. Blaschko and C. H. Lampert. Correlational spectral clustering. In *Proc. of CVPR '08*, pages 1–8, 2008.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[6] C. Boutsidis and E. Gallopoulos. Svd based initialization: A head start for nonnegative matrix factorization. *Pattern Recognition*, 41(4):1350–1362, 2008.

[7] E. Bruno and S. Marchand-Maillet. Multiview clustering: A late fusion approach using latent models. In *Proc. of SIGIR '09*, pages 736–737, 2009.

[8] C. Carpineto, S. Osiński, G. Romano, and D. Weiss. A survey of web clustering engines. *ACM Computing Surveys*, 41(3):17, 2009.

[9] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan. Multi-view clustering via canonical correlation analysis. In *Proc. of ICML '09*, pages 129–136, 2009.

[10] P. Das, R. Srihari, and Y. Fu. Simultaneous joint and conditional modeling of documents tagged from two perspectives. In *Proc. of CIKM '11*, pages 1353–1362, 2011.

[11] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1-2):143–175, 2001.

[12] C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix tri-factorizations for clustering. In *Proc. of KDD '06*, pages 126–135, 2006.

[13] C. H. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proc. of SDM '05*, pages 606–610, 2005.

[14] K. Filippova and K. B. Hall. Improved video categorization from text metadata and user comments. In *Proc. of SIGIR '11*, pages 835–842, 2011.

[15] E. Gaussier and C. Goutte. Relation between plsa and nmf and implications. In *Proc. of SIGIR '05*, pages 601–602, 2005.

[16] D. Greene and P. Cunningham. A matrix factorization approach for integrating multiple data views. In *Proc. of ECML/PKDD '09*, pages 423–438, 2009.

[17] A. Hindle, J. Shao, D. Lin, J. Lu, and R. Zhang. Clustering web video search results based on integration of multiple features. *World Wide Web*, 14(1):53–73, 2011.

[18] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177–196, 2001.

[19] P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[20] C.-F. Hsu, J. Caverlee, and E. Khabiri. Hierarchical comments-based clustering. In *Proc. of SAC '11*, pages 1130–1137, 2011.

[21] M. Hu, A. Sun, and E.-P. Lim. Comments-oriented document summarization: understanding documents with readers' feedback. In *Proc. of SIGIR '08*, pages 291–298, 2008.

[22] F. Jing, C. Wang, Y. Yao, K. Deng, L. Zhang, and W.-Y. Ma. Igroup: web image search results clustering. In *Proc. of MM '06*, pages 377–384, 2006.

[23] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[24] A. Kumar, P. Rai, and H. D. Iii. Co-regularized multi-view spectral clustering. In *Proc. of NIPS '11*, pages 1413–1421, 2011.

[25] T. Kuzar and P. Navrat. Slovak blog clustering enhanced by mining the web comments. In *Proc. of WI-IAT '11*, pages 293–296, 2011.

[26] A. N. Langville, C. D. Meyer, R. Albright, J. Cox, and D. Duling. Initializations for the nonnegative matrix factorization. In *Proc. of KDD '06*, pages 23–26, 2006.

[27] T. Lappas, K. Punera, and T. Sarlos. Mining tags using social endorsement networks. In *Proc. of SIGIR '11*, pages 195–204, 2011.

[28] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[29] B. Li, S. Xu, and J. Zhang. Enhancing clustering blog documents by utilizing author/reader comments. In *Proc. of ACM-SE '07*, pages 94–99, 2007.

[30] C. Liu, H.-c. Yang, J. Fan, L.-W. He, and Y.-M. Wang. Distributed nonnegative matrix factorization for web-scale dyadic data analysis on mapreduce. In *Proc. of WWW '10*, pages 681–690, 2010.

[31] D. Liu, X.-S. Hua, L. Yang, M. Wang, and H.-J. Zhang. Tag ranking. In *Proc. of WWW '09*, pages 351–360, 2009.

[32] J. Liu, C. Wang, J. Gao, and J. Han. Multi-view clustering via joint nonnegative matrix factorization. In *Proc. of SDM '13*, pages 252–260, 2013.

[33] B. Long, S. Y. Philip, and Z. M. Zhang. A general model for multiple view unsupervised learning. In *Proc. of SDM '08*, pages 822–833, 2008.

[34] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press Cambridge, 2008.

[35] F. Pedregosa, G. Varoquaux, Gramfort, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[36] D. Ramage, P. Heymann, C. D. Manning, and H. Garcia-Molina. Clustering the tagged web. In *Proc. of WSDM '09*, pages 54–63, 2009.

[37] D. Seung and L. Lee. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13:556–562, 2001.

[38] J. Wang, H. Zeng, Z. Chen, H. Lu, L. Tao, and W.-Y. Ma. Recom: reinforcement clustering of multi-type interrelated data objects. In *Proc. of SIGIR '03*, pages 274–281, 2003.

[39] Y.-X. Wang and Y.-J. Zhang. Nonnegative matrix factorization: A comprehensive review. *Knowledge and Data Engineering, IEEE Transactions on*, 25(6):1336–1353, 2013.

[40] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proc. of SIGIR '03*, pages 267–273, 2003.

[41] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma. Learning to cluster web search results. In *Proc. of SIGIR '04*, pages 210–217, 2004.

[42] M. Zitnik and B. Zupan. Nimfa: A python library for nonnegative matrix factorization. *Journal of Machine Learning Research*, 13:849–853, 2012.