2017

A Power Iteration Based Co-Training Approach to Achieve Convergence for Multi-View Clustering

Pavankalyan Yallamelli

Wright State University, yallamelli.2@wright.edu

Follow this and additional works at: https://corescholar.libraries.wright.edu/etd_all

Part of the Computer Sciences Commons

Repository Citation


https://corescholar.libraries.wright.edu/etd_all/1792

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact library-corescholar@wright.edu.
A Power Iteration Based Co-Training Approach to Achieve Convergence for Multi-View Clustering

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

By

Pavankalyan Yallamelli
B.Tech, Jawaharlal Nehru Technological University, 2013

2017
Wright State University
I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY PAVANKALYAN YALLAMELLI ENTITLED A POWER ITERATION BASED CO-TRAINING APPROACH TO ACHIEVE CONVERGENCE FOR MULTI-VIEW CLUSTERING BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE.

_____________________________________
Amit P. Sheth, Ph.D.
Thesis Director

_____________________________________
Mateen M. Rizki, Ph.D.
Department Chair

Committee on
Final Examination

_____________________________________
Amit P. Sheth, Ph.D.

_____________________________________
Keke Chen, Ph.D.

_____________________________________
Brandon Minnery, Ph.D.

_____________________________________
Barry Milligan, Ph.D.
Interim Dean of Graduate School
ABSTRACT

YALLAMELLI, PAVANKALYAN. M.S., Department of Computer Science and Engineering, Wright State University, 2017. A Power Iteration Based Co-Training Approach to Achieve Convergence for Multi-View Clustering.

Collecting diversified opinions is the key to achieve "the Wisdom of Crowd". In this work, we propose to use a novel multi-view clustering method to group the crowd so that diversified opinions can be effectively sampled from different groups of people.

Clustering is the process of dividing input data into possible subsets, where every element (entity) in each subset is considered to be related by some similarity measure. For example, a set of social media users can be clustered using their locations or common interests. However, real-world data is often best represented by multiple views/dimensions. For example, a set of social media users have a friend/follower network as well as a conversation network (different from a follower network).

Multiple views enable a better understanding of data by improving knowledge accuracy through cross verification across different views; it also improves the performance by integrating multiple views. Multi-view clustering enables this. Clustering quality, clustering agreement (consensus) and scalability are the three essential qualities for achieving higher correspondence between the clusters and the real underlying groups in multi-view clustering. Existing algorithms either lack
scalability or achieve cluster convergence (consistent clusters across the views) very slowly. Most of the existing and recent multi-view clustering algorithms make use of spectral clustering. Spectral clustering which ensures higher accuracy is computationally costly because of eigenvector computation. To address this gap, in this paper we propose a clustering mechanism based on a co-training approach that achieves the three qualities.

The two main contributions of our work are as follows: (1) a learning method using power-iteration clustering for clustering a single data view, and (2) an efficient and scalable update method that uses the cluster label information for updating other data views iteratively to achieve convergence (clustering agreement) and cluster quality.

The proposed method is evaluated on two real-world datasets to show that it outperforms existing approaches in terms of clustering quality and consensus. We evaluate the clustering quality in the context of a Wisdom of Crowds application. Specifically, we use clustering to identify groups of similar users (crowd members) based on their social media conversations (Tweets) related to a particular topic, in this case, fantasy sports (Fantasy Premier League soccer in particular). We then form virtual groups of diverse and non-diverse users based on the clusters identified. Our results show that diverse crowds outperform non-diverse crowds in a typical fantasy sports task (picking a team captain), consequently validating our cluster qualities.
# Table of Contents

1. Introduction .........................................................................................................................1

2. Background ..........................................................................................................................8
   2.1. Multi-View Clustering .....................................................................................................8
       2.1.1. Overview ..............................................................................................................8
       2.1.2. Importance and Advantages ...............................................................................8
       2.1.3. Related Work ......................................................................................................9
       2.1.4. Multi-view Clustering Categories ......................................................................9
   2.2. Power Iteration Clustering .............................................................................................12
       2.2.1. Overview ............................................................................................................12
       2.2.2. Importance and Advantages .............................................................................13
       2.2.3. Algorithm ..........................................................................................................13

3. Co-training based multi-view clustering ..............................................................................15

4. Approach .............................................................................................................................18
   4.1. Co-training with Power Iteration clustering .................................................................18

5. Experimental Results ..........................................................................................................24
   5.1. Evaluation Metrics .......................................................................................................24
   5.2. Synthetic Graph Dataset ............................................................................................25
   5.3. Twitter UK Politics Dataset .......................................................................................28
   5.4. Wisdom of Crowd Dataset ........................................................................................30
       5.4.1. Experimental Setup ..........................................................................................31

6. Related Work .......................................................................................................................35

7. Conclusion ............................................................................................................................37

Bibliography .............................................................................................................................38

Appendix A ................................................................................................................................41
List of Figures

Figure 1. Individual vs. Non-diverse vs. Diverse crowd. ................................................................. 2

Figure 2. Co-training style algorithm ................................................................................................. 10

Figure 3. Sketch Map for Multi-kernel learning .................................................................................. 11

Figure 4. Sketch map of subspace learning for multi-view data ......................................................... 12

Figure 5. Workflow of Co-training based spectral clustering approach .............................................. 16

Figure 6. Example for Co-training based Spectral Clustering ............................................................ 17

Figure 7. Architecture for proposed method ......................................................................................... 19

Figure 8. Workflow of Co-training with Power iteration clustering .................................................... 21

Figure 9. Example of Co-training based power iteration clustering using new update method ............... 22

Figure 10. NMI scores in different views vs. the number of iterations of co-trained spectral clustering and power iteration clustering for Synthetic data. Each view is represented by a different color .................................................................................................................. 27

Figure 11. NMI and Variance of co-trained spectral clustering and power iteration clustering for Synthetic data ..................................................................................................................................... 27

Figure 12. NMI scores in different views vs. number of iterations of co-trained spectral clustering and power iteration clustering for Twitter data ................................................................................. 29

Figure 13. NMI and Variance of co-trained spectral clustering and power iteration clustering for Twitter data ..................................................................................................................................... 29

Figure 14. Captain scores for diverse and non-diverse crowd vs. crowd size for player’s view .................. 33

Figure 15. Captain scores for diverse and non-diverse crowd vs. crowd size for team view .................... 34
Figure 16. Captain scores for diverse and non-diverse crowd vs. group size for both the views…………………………………………………………………………………………………………………………34
List of Tables

Table 1. Results of Cohen’s kappa for the 3 datasets ...................................................... 25
Table 2. Clustering performance on Synthetic Dataset ...................................................... 26
Table 3. Clustering performance on Twitter Dataset ......................................................... 28
Acknowledgment

This work was partially supported by Army Research Office Grant No. W911NF-16-1-0300.

I would firstly like to thank my advisor Dr. Amit Sheth. Not only has he been a great advisor to me, but a great mentor as well. He provided me with constant support both personally and academically through my graduate journey. I have learnt a lot from the interactions I have had with him and will carry these valuable lessons throughout my life. I would like to thank Shreyansh Bhatt for being a wonderful mentor and a great friend. I am grateful to him for the insightful discussions, brainstorming sessions and for all his support and guidance.

I would also like to thank my committee members Dr. Keke Chen and Dr. Brandon Minnery for being on my panel and reviewing my work. I would also like to thank Dr. Prasad, Sarasi and Vinh for guiding me on my initial projects; this prepared me for the latter part of my graduate program. Being a part of Kno.e.sis has made this process a lot of fun and I would like to thank the entire Kno.e.sis family for it.

This acknowledgment would be incomplete without thanking my dear friends Nishita, Shalini, Roolteja, Yamini and my parents for their help, hard-work, and support.
My graduate program has been a fulfilling one, and I am excited for whatever is in store next.

Last but not the least, I am thankful to my parents Jaya Lakshmi and Y.N.S.C Bose, for all their continued faith in me. This thesis is dedicated to you.
1. Introduction

The "wisdom of crowds" (WoC) effect is one of the best-known examples of collective human intelligence. WoC is a phenomenon in which an aggregated judgment of a group of people is shown to be more accurate (on average) than that of any one individual within the group. A well-known quote attributed to Ken Blanchard is: “None of us is as smart as all of us.” [1]

James Surowiecki’s New York Times bestseller The Wisdom of Crowds states that “a diverse collection of independently-deciding individuals is likely to make certain types of decisions and predictions better than individuals or even experts” [2]. Figure 1 illustrates this basic idea that a crowd can be smarter than an individual, and a diverse crowd is even smarter. Various forms of domain-specific and/or domain independent diversity can affect WoC. It is interesting to ask whether a domain-independent diversity measure based on (for example) geolocation versus a domain-specific measure based on (for instance) crowd members’ subject matter knowledge or a combination of both helps create smarter groups. This has led to an interest in studying and understanding multiple diversity measures to form an intelligent crowd. These experiments are mostly performed in a lab environment where an experimenter gathers a group of subjects and asks them to perform a
certain task, such as a prediction task[3]. Social media makes such data available to users. It is also interesting to study the wisdom of crowd based on diversity computed from social media [4].

Figure 1: Individual vs Non-diverse vs Diverse crowd.

In a recent study, Bhatt et al. [4] showed that social media data analysis could be used to measure crowd diversity and to predict differences in performance between diverse and non-diverse crowds. Social media data is considered to have various dimensions. For example, consider diversity based on interest in a soccer team, where a set of users is involved in a soccer-related discussion. In this example, diversity can be measured based on team mentions and player mentions. Here we can consider the *team mentions* as one dimension (view) and *player mentions* as another dimension (view). Given the essential role of diversity in WoC effects, there
is a surprising lack of research on how to measure the diversity of a crowd at scale [4]. Although the existing methodology of WoC is efficient, such an approach usually attributes superiority of the crowd interests over individual judgment. In other words, it prioritizes the opinions and interests of crowd averages over an individual's point of view. To overcome such a gap, we need a mechanism to measure crowd diversity.

In this paper, we present a novel clustering method which facilitates measuring crowd diversity. Based on members’ social media (Twitter) communications, we examined whether it is possible to extract measures of crowd diversity and checked if those measures can drive the selection of wiser crowds. Our principal assumption is that content of a person's communications involving a particular judgment problem carries the knowledge of what the person knows about the issue and those factors and knowledge sources he or she believes are most appropriate to formulate a judgment. We identified tweets related to Fantasy Premier League (FPL) fantasy.premierleague.com as a promising data source for evaluating our hypothesis. For the domain of fantasy sports, we aimed to come up with a technique that identifies crowds (a set of fantasy team owners) that are diverse based on their tweet content. We show that such diverse crowds demonstrably outperform less diverse crowds.

Most Twitter users post their profile information, which can be utilized to match their Twitter record with their other publicly available social data, such as their
publicly posted fantasy sports activity. Thus we can match the tweets of fantasy team owners with their judgments (i.e., their team picks) and the outcomes of those judgments (i.e., their scores) [4].

One approach to forming diverse crowds is to identify subgroups of similar users and to select users from multiple subgroups. We identify subgroups of similar users using a clustering mechanism. Clustering is the process of dividing input data into possible subsets where elements in each subset are considered related by some similarity measure. Existing work suggests that a clustering approach could be used to generate more diverse, wiser crowds [4]. The most common approach to clustering involves describing an entity (user) by a feature vector. For example, one feature can be a user description based on soccer teams that a user is interested in while another feature can be a user description based on soccer players that a user is interested in. In this example, users could be clustered based on their interest in a particular team or their interest in a particular player. Since we do not know the importance of a feature in the clustering process, we use multi-view clustering to find a similar set of users.

For a multi-dimensional clustering problem, one can create diverse and non-diverse crowds based on the clustering technique or quality of clusters. Clustering algorithms are assessed mostly on three criteria -- time complexity, clustering quality, and consensus. Multi-view clustering techniques are mostly categorized as
one of the following: subspace algorithm; multiple kernel learning; or co-training [5].

In this paper, we develop a co-training approach for cluster identification. Essentially, co-training style algorithms train alternately to maximize a mutual agreement on two distinct views of the data. One of the recent well-known co-training based clustering approaches is based on spectral clustering [6], which uses spectral embedding for co-training, i.e., to update views. Since spectral clustering [7] can be a bottleneck for large graphs, we explore the use of a more efficient spectral based clustering method, power iteration clustering [8]. Power iteration clustering (PIC) does not provide spectral embeddings as spectral clustering. Specifically, PIC does not depend on computing all the eigenvalues. Hence, we cannot use spectral embedding for co-training. Moreover, such a spectral embedding based co-training fails to achieve clustering consensus across views, which is the most important quality in multi-view clustering.

Achieving a consensus is essential for any co-training based multi-view clustering algorithm. In a co-training based approach, each view generates a clustering result in every iteration. If the generated results of a view do not agree with other views, it is hard for someone to select clustering results for a view randomly. For example, consider two views with five nodes (say 1,2,3,4,5). If clustering result of the first view is (\{1,3,5\}, \{2,4\}), and the clustering result of the second view is (\{1,2,5\}, \{3,4\}), then it is hard for someone to select one result out of these two. Instead if
the clustering of both the views results in the same clusters \{(1,3,5), (2,4)\}, then it is easy to select one. Hence, the consensus is a significant property for multi-view clustering.

In this work, we propose a novel approach that integrates the co-training approach with PIC using a new view update mechanism which achieves fast view agreement and works more efficiently for multi-view clustering. Our new update mechanism considers the clustering labels in co-training rather than using spectral embedding, which helps in achieving convergence. We use two datasets to evaluate the clustering quality and clustering consensus and show that our approach achieves better normalized mutual information (NMI) compared to existing co-training based multi-view clustering approaches. We then use the proposed co-training based multi-view clustering method on a Twitter user dataset [4] and show that the resulting cluster structure can be used to select more diverse, and hence wiser, crowds. Our results show that such a diverse crowd performs better at a captain prediction task than a non-diverse crowd.

The main contributions are as follows:

1. Scalable co-training based multi-view clustering algorithm using PIC which in most cases is faster than state of the art.

2. Cluster label based update for co-training which achieves convergence.
3. Detailed experiment and comparison of the proposed approach with existing multi-view clustering techniques.


5. Clustering-based measures of crowd diversity which can be used to guide the selection of more diverse, hence wiser, crowds.

We have organized the paper as follows: we begin with Section 2, background description of multi-view clustering and PIC. Then we move on to discuss existing co-training based multi-view clustering in Section 3. Section 4 explains the clustering agreement method we propose in our work. Section 5 presents the experimental results on real, synthetic and WoC datasets and compares them with existing approaches. This will be followed by discussions of related work and conclusions in Section 6 and 7 respectively.
2. **Background**

2.1. **Multi-View Clustering**

2.1.1. **Overview**

Many real-world datasets can be described using multiple dimensions/features. For example, consider a set of web pages. The hyperlink network between these pages conveys a relationship between them and can serve as a measure of similarity between pages. The textual and/or multimedia content contained in the web pages also serves as a measure of similarity; i.e., two pages can be compared based on their content. The similarity computed based on multiple relationships between these data points are also referred as *views*.

2.1.2. **Importance and Advantages**

Traditionally it was understood that one particular subset would be sufficient for data mining, and multiple views were often regarded as redundant. However, research has now illustrated that these multiple views are often complementary [6] and help gain a better understanding of the data structure. Multiple view learning has two advantages: a better performance can be obtained by integrating the multiple views rather than a single view, and the accuracy of the knowledge
produced can be cross-verified from multiple views. Hence there is a need for multi-view clustering to handle multi-view data.

2.1.3. Related Work

Over the recent years, many successful multi-view learning methods have been introduced (e.g., co-training [9], co-EM [10] and co-regularized multi-view spectral clustering algorithms [24]). After analyzing several algorithms, we observed that the most crucial principle for multi-view clustering is the consensus. Consensus tries to maximize the agreement between the clusters in distinct views.

2.1.4. Multi-view Clustering Categories

As described above, multi-view clustering offers advantages over traditional clustering methods in terms of performance and cross-verification. For these purposes, there has been burgeoning interest in multi-view clustering approaches. These approaches can be classified into three categories: 1) subspace; 2) multiple kernel learning, and 3) co-training.[5]

2.1.4.1. Co-training

Co-training is one of the earliest techniques for multi-view learning proposed for semi-supervised learning problems by Blum & Mitchell in 1998 [11]. The co-training approach trains to maximize the agreement on two distinct views of the
data. Three main assumptions that co-training algorithms relies on are: (a) sufficiency - each view is sufficient for classification on its own; (b) compatibility - the target function of both views predicts the same labels for co-occurring features with a high probability; and (c) conditional independence - views are conditionally independent given the label [5].

Co-training maximizes the agreement between the views of the unlabeled data by training it alternatively on distinct views. It minimizes the error on the training set and maximizes the agreement on unlabeled data by eventually producing one right classifier for each specific view. The co-training for multi-view clustering works under an assumption that a point or vertex has to be assigned to the same cluster by each view. Hence, co-training for clustering can be done according to cluster affiliation.

![Co-training style algorithm](image)

Figure 2: Co-training style algorithm [5].
2.1.4.2. Multi-kernel Learning

In the multiple kernel learning (MKL) approach, individual kernels are generated for each view and merged with a kernel-based method to form a unified kernel. In this approach, all the kernels are merged when training the dataset. In MKL, kernels naturally correspond to distinct views, and merging these kernels improves the learning performance. In MKL different kernels may correspond to different similarities.

![Figure 3: Sketch Map for Multi-kernel learning [5].](image)

2.1.4.3. Subspace

Subspace is a well-known approach for multi-view clustering used to analyze the correlation between various distinct views and has many different applications. The Subspace methods try to obtain a shared representation of all the views by obtaining
an appropriate subspace assuming that input views are generated from a latent subspace [5]. The generated latent subspace has lower dimensionality than any other input views. Hence, subspace learning mostly depends on dimensionality reduction. Therefore, this kind of approach can be used in prior combination of multiple views. Since the information exchange happens at the feature level, the merging does not incorporate graph structure from multiple views and can result in poor clustering quality for real-world datasets.

![Figure 4: Sketch map of subspace learning for multi-view data [5].](image)

2.2. Power Iteration Clustering

2.2.1. Overview

Power iteration clustering (PIC) is a semi-supervised learning algorithm, which works on matrix-vector multiplication [8]. PIC finds a very low-dimensional
embedding of a dataset using truncated power iteration on a normalized pairwise similarity matrix of the data. In other words, PIC identifies a combination of dominant eigenvectors of a matrix that reveals the underlying clustering structure of data.

2.2.2. Importance and Advantages

PIC has gained prominence due to its mathematical framework and its capability to deliver good results with arbitrarily shaped clusters, which is otherwise a shortcoming with several other clustering algorithms such as k-means and spectral clustering algorithms [8]. It is also proven that PIC is faster than traditional Ncut implementation [8]. Hence, we adopted the PIC technique to solve multi-view clustering problems.

2.2.3. Algorithm

**Algorithm 1 The PIC algorithm**

**Input:** A row-normalized affinity matrix $W$ and the number of clusters $k$  
Pick an initial vector $v^0$

**repeat**

- Set $v^{t+1} \leftarrow \frac{Wv^t}{\|Wv^t\|_1}$ and $\delta^{t+1} \leftarrow |v^{t+1} - v^t|$.
- Increment $t$

**until** $|\delta^t - \delta^{t-1}| \approx 0$

Use k-means to cluster points on $v^t$

**Output:** Clusters $C_1, C_2, \ldots, C_k$

Algorithm 1: Power Iteration Clustering [8]
Here we briefly outline the PIC algorithm (see [8] for a detailed introduction). Given a row-normalized affinity matrix and an initial vector, PIC iteratively performs the update until it generates the vector that is most similar to the vector that is used to update.
3. Co-training based multi-view clustering

In this section, we provide a summary of co-training based multi-view clustering with existing methods and their drawbacks. The basic idea of co-training algorithms is that the clustering results from one view are used to constrain the similarity for the other views. Several previous attempts related to co-training [6,9,12] have achieved better clustering results. All these algorithms use spectral clustering, which handles irregular cluster shapes and thus is good for graph clustering. The latest co-training based multi-view clustering approach [6] uses spectral embedding from one view to update other views. The spectral embedding of a view represents an underlying clustering structure since this low-dimensional embedding is the one which is used to generate clusters. However, the spectral embedding does not represent clear boundaries between clusters; hence it affects co-training algorithm to achieve convergence. Moreover, spectral clustering used in the whole process has issues with scalability and runtime. Figure 5 shows a single iteration of co-training based spectral clustering approach [6]. Matrix A and B are affinity matrices which represent view1 and view2 respectively. In spectral clustering, top k eigenvectors are computed from normalized graph Laplacian of each view’s affinity matrix.
Figure 5: Workflow of co-training based spectral clustering approach

Once the k eigenvectors of each view are generated, they are then multiplied with its transpose to generate a n*n matrix X for the first view and Y for the second view. Matrix X and Y represent the clustering affiliation (spectral embedding) of the first view and second view respectively. Co-training algorithms train in a way that spectral embedding of one view is multiplied to affinity matrix of the second view, i.e., X is multiplied to B and Y is multiplied with A for the second iteration. This process will be carried out for a certain number of iterations. However, spectral embedding does not provide the precise boundaries between the clusters because eigenvalues can be negative and can be spread for future iterations. Moreover, it is very high in time complexity because of eigenvector computation, and a lot of unnecessary edges will be added between the nodes which makes the graph thick. Example 1 illustrates the disadvantages of spectral clustering with co-training.
The graph (Figure 6) depicts spectral clustering with co-training. In this example, there are five nodes A, B, C, D, E and there are edges between each node with some value on it in view1 and view2. It is shown that after running co-training with spectral clustering for one iteration, new edges BE and CD are created with negative values associated with it in view1. Meanwhile, there is no edge between DE in both the views, but after the first iteration, a new edge has been added in both the views with some value associated with it.

Figure 6: Example for Co-training based Spectral Clustering
4. Approach

4.1. Co-training with Power Iteration clustering

In this section, we develop a new co-training framework to carry out multi-view PIC. For any multi-view clustering, it is essential to achieve good quality clusters and consensus across all the views. Consensus can be defined as the clustering agreement across all the views, i.e., each view generates the same clustering result at the end of the co-training algorithm. Existing algorithms either lack in achieving consensus or clustering quality or scalability. Hence, we approached this problem with three broad aims – achieving consensus, scalability, and clustering quality.

First, we introduce a new method to update an affinity matrix of an augmented view using the cluster labels of all the other views. This ensures that, unlike spectral embedding, we do not influence the affinities between the vertices that belong to different clusters. This update method helps to achieve consensus by establishing a guided co-training procedure and consequently converges to a more efficient affinity matrix for PIC in every iteration. This update method works on the assumption that if two points present in the same cluster in the maximum number of views, it should be so in all the views. On the other hand, if two points belong to different clusters, it should be so in all the views. We use the clustering labels of
each view which are generated from k-means algorithms to update the views for the next iteration instead of the eigenvector. The proposed update method adds the weight between the nodes if the nodes belong to the same cluster in rest of the views.

The update view method is defined as

\[ A_v = A_v + \alpha \sum_{i=1}^{N} C_i \] where \( \alpha = \frac{1}{v} \), \( A_v \) represents the affinity matrix of a view \( V = \{1, 2, 3, \ldots, n\} \), \( \alpha \) represents the constant and \( C_i \) represent the clustering label information.

![Figure 7: Architecture for proposed method](image)

The second most critical component of the proposed approach is scalability. PIC has gained prominence in recent years due to the capability of delivering good results even with arbitrarily shaped clusters, which is otherwise a shortcoming with
several other clustering algorithms. When compared to spectral clustering, the cost (in space and time) of explicitly calculating eigenvectors is replaced by that of a small number of matrix-vector multiplications. Hence, we adopted PIC technique to solve multi-view clustering problem. PIC generates the top eigenvector of a given matrix, unlike spectral clustering which generates top k eigenvectors.

Figure 7 demonstrates the idea of co-training with PIC using the proposed update method. Given the affinity matrix of v views say $A_1, A_2, \ldots, A_v$ where $A_v$ is n*n affinity matrix of view v and $v = \{1,2, \ldots, v\}$. And $E_1, E_2, \ldots, E_v$ represent the eigenvectors of corresponding views which are computed using PIC. We apply the k-means algorithm on this eigenvector to generate the cluster labels $C_1, C_2, \ldots, C_v$. We check the error rate between all the cluster labels using NMI. If the error rate is below the given threshold, the co-training algorithm terminates else affinity matrix of each view will be computed using proposed view update method, and these updated views are used as the input for the subsequent iteration.
Figure 8: Workflow of Co-training with Power iteration clustering.

The below figure shows how the update method works. We considered the same input graph (Figure 9) to demonstrate the workflow of the proposed algorithm. We can make two observations from this example, unlike spectral clustering, our method does not add invalid edges such as DE, and thus it does not make the graph dense.
The stepwise implementation of our co-training with PIC algorithm is defined in Algorithm 2. Here the inputs are views which are defined using the similarity matrices. We compute the normalized pairwise similarity matrix of data points, which is represented by $A_1, A_2, ..., A_v$. Next, we identify the total number of views we want to represent the data in. This is denoted by $v$; $k$ represents the total number of clusters and $\text{iter}$ represents the number of total iterations for the clustering algorithm. For the total number of predefined views, we repeatedly apply the PIC algorithm to identify the dominant eigenvector for each of the similarity matrices.
Algorithm 2: A Co-training Approach for Multi-view Power Iteration Clustering Algorithm

**Input**: Similarity matrix $A_1, A_2, A_3, \ldots, A_v$, where $A \in R_{n \times n}$, 
$v = \text{number of views}$,  
$k = \text{number of clusters}$,  
$\text{iter} = \text{number of iterations}$

**Output**: Assignments to $k$ clusters

for $i = 1 \rightarrow v$ do  
    $[E_v, C_v] = PIC(A, k)$ 
end

for $i = 1 \rightarrow \text{iter}$ do 
    Construct $R_v \ \forall v$ such that $R_{vu}$ $\leftrightarrow$ two data points belong to the same cluster in $C_v$ 
    
    $R = \sum_{i=1}^{v} R_i$
    
    for $j = 1 \rightarrow v$ do 
        $y_j = A_j + \frac{\text{avg}(A_j)}{v} \cdot [R - R_j]$ 
        $[E_v, C_v] = PIC(A_j, k)$ 
    end
end

Algorithm 2: A power iteration based co-training approach for multi-view clustering

Some of the other advantages include the fact that PIC clustering can be replaced by any other efficient algorithm, and also one of the views can be prioritized by increasing the value of alpha in the update method.
5. Experimental Results

In this section, we compare our approach with the number of baseline methods like the single view, feature concatenation and co-training with spectral clustering on three different datasets to show the convergence of our approach. Three data sets are-- a synthetic dataset, a real-world dataset, and wisdom of crowd dataset. Each dataset has a different number of views and a different number of data points (nodes).

5.1. Evaluation Metrics

We evaluate the clustering results using different measures like NMI and Cohen's kappa coefficient and Monte Carlo Simulation. All measures mentioned above return a value between 0 and 1, and the higher the value, the better the clustering results. The Cohen’s kappa is used to measure the agreement between every possible pair of views. It represents the degree of accuracy and reliability. If Cohen’s kappa results in 1, then it indicates absolute agreement between the views, and if it results in 0, then it indicates that any agreement is entirely due to chance. Kappa can result in any adverse value, although we are interested only in values between 0 and 1. Results reported in Table 1 are the average value of Cohen’s kappa between every possible view pair of three datasets (described below). The results show that our approach outperforms co-training with spectral clustering.
and can achieve absolute agreement for the synthetic dataset and Twitter UK politics dataset.

<table>
<thead>
<tr>
<th>Dataset/method</th>
<th>Co-training with Spectral Clustering</th>
<th>Our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic dataset</td>
<td>0.2657</td>
<td>1</td>
</tr>
<tr>
<td>Twitter UK politics dataset</td>
<td>0.08</td>
<td>1</td>
</tr>
<tr>
<td>Wisdom of Crowd dataset</td>
<td>0.053</td>
<td>0.957</td>
</tr>
</tbody>
</table>

Table 1: Results of Cohen’s kappa for the 3 datasets

5.2. Synthetic Graph Dataset

This dataset is introduced by Kumar [6]. It consists of three different views with 1000 nodes. The synthetic dataset is created/synthesized to represent complementary information about 1000 data-points. Since each view has complementary information, clustering consensus is not easy to achieve, and best clusters are found when consensus is achieved.

We used this synthetic dataset to show how our algorithm achieves consensus. We compared our approach with the number of baseline methods and co-training for spectral clustering method. Firstly, we applied Cohen's kappa coefficient to evaluate the clustering agreement between the views (Table 1). Since the existing approach did not mention the
stopping criteria of their algorithm, we report the results obtained in the fifth iteration; however, our algorithm achieved convergence in the fourth iteration.

Second, we applied NMI to evaluate the clustering quality. We compared the results of both the approaches with ground truth values. Since the existing approach did not mention the stopping criteria and is not able to generate homogeneous clustering results, we report results obtained in the fifth iteration of a randomly selected view (Table 2).

<table>
<thead>
<tr>
<th>Method</th>
<th>F-Score</th>
<th>Precision</th>
<th>Recall</th>
<th>Entropy</th>
<th>NMI</th>
<th>Adj-RI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Co-trained spectral</td>
<td>0.94</td>
<td>0.94</td>
<td>0.93</td>
<td>0.19</td>
<td>0.797</td>
<td>0.87</td>
</tr>
<tr>
<td>Our Approach</td>
<td>0.99</td>
<td>0.99</td>
<td>0.989</td>
<td>0.04</td>
<td>0.95</td>
<td>0.98</td>
</tr>
</tbody>
</table>

Table 2: Clustering performance on Synthetic Dataset

The graph below (Figure 10) clearly explains the comparison of convergence between the existing approach and our approach. In Figure 11 it is evident that our approach converges to an agreed view with high NMI and almost same clustering results.
Figure 10: NMI scores in different views vs. the number of iterations of co-trained spectral clustering and power iteration clustering for Synthetic data. A different color represents each view.

Figure 11: NMI and variance of co-trained spectral clustering and power iteration clustering for Synthetic data

- Higher average NMI
- Almost identical final views (low variance)
5.3. Twitter UK Politics Dataset

This dataset is introduced by Greene and Derek [23]. It contains seven views (followed by, follows, retweeted by, retweets, tweets, mentioned by, mentions) with 1200 users (nodes). Each of the seven views brings complementary information, and clusters found based on all the information are likely to be more robust as well as more accurate.

Firstly, we applied Cohen's kappa coefficient to evaluate the clustering agreement between the views (Table 1). Since the existing approach did not mention the stopping criteria of their algorithm, we report the results obtained in the fifth iteration; however, our algorithm achieved convergence in the fourth iteration.

Second, we applied NMI to evaluate the clustering quality. We compared the results of both approaches with ground truth values. Since the existing approach did not specify the stopping criteria and was not able to generate consistent clustering result, we report results obtained in the fifth iteration of a randomly selected view in table 3.

<table>
<thead>
<tr>
<th>Method</th>
<th>F-Score</th>
<th>Precision</th>
<th>Recall</th>
<th>Entropy</th>
<th>NMI</th>
<th>Adj-RI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Co-trained spectral</td>
<td>0.906</td>
<td>0.870</td>
<td>0.947</td>
<td>0.34</td>
<td>0.801</td>
<td>0.84</td>
</tr>
<tr>
<td>Our Approach</td>
<td>0.996</td>
<td>0.996</td>
<td>0.998</td>
<td>0.02</td>
<td>0.986</td>
<td>0.994</td>
</tr>
</tbody>
</table>

Table 3: Clustering performance on Twitter Dataset
The below graph (Figure 12) results show that all the views converge to same clustering results and also achieve high NMI. Figure 13 show that our algorithm is able to achieve higher average NMI with almost identical clustering results.

Figure 12: NMI scores in different views vs. number of iterations of co-trained spectral clustering and power iteration clustering for Twitter data

Figure 13: NMI and Variance of co-trained spectral clustering and power iteration clustering for Twitter data
5.4. Wisdom of Crowd Dataset

Here we are applying our algorithm to study the effects of diverse users (crowd members) in a wisdom-of-crowd application, where a group of individuals participating in a study can most certainly outperform the judgment of the single expert. For example, if the weight of an item has to be judged, the average estimate of the group will be more accurate than a single judgment from an expert. By using a diverse group, we get a more accurate judgment. These results tend to be more unbiased since the group is not a focused group of people specializing in one domain.

Gathering information from crowd-sourced applications is a widely used technique for knowledge acquisition tasks. We used this insight to evaluate the performance of our algorithm and to compare the system to the state-of-the-art. Crowdsourcing is a cost-effective and reliable approach to efficiently distribute a task among a potentially diverse and large group of contributors, and our results show that diverse crowds sampled using clusters defined by our algorithm outperform non-diverse crowds.

For the experiments discussed in this paper, we used a publicly available judgment dataset in the form of FPL player picks. This dataset consists of 2M Tweets (soccer-related tweets ~ 1M, FPL-specific tweets ~ 90k) based on which diversity is computed. FPL is an online game. Here each user will be given an initial set of points. Initially, users will select the players and captain with the points that they have. Once the game is performed, each user will be rewarded by points depending on the player’s performance in the game. Users are
motivated to select a captain that gives them the best reward [4]. Our goal is to predict player performance based on users’ opinions.

We collected the tweets via Twitter streaming API using keywords related to FPL of English Premier League season 2016-17 from August 2016 to November 2016 corresponding to the first four months (25 weeks) of the season. We considered tweets related to teams mentions as one view and tweets related to players mentions as another view. Using these tweets, we obtained the names of Twitter users. We extracted the captain pick data from the FPL website by matching the Twitter users name and username on the FPL website.

5.4.1. Experimental Setup

Our experimental setup was carried out as follows: initially created a similarity matrix for each view, that is for tweets which have mentions of FPL players and tweets with mention of the teams. This matrix was generated using the Word2vec based diversity. Word2Vec measure is used by most state-of-the-art systems to compute the semantic similarity between words using average pairwise cosine distance. Word2Vec has been used for natural language processing tasks, including hashtag prediction in tweets [16], sentence completion as reported in work by Godin, Fréderic, et al. [14], and part-of-speech tagging [15]. Research suggests that Word2Vec can be applied to social data as well. In work by De Boom, Cedric et al. and Wijeratne Sanjaya, et al., the authors show that it is possible to
identify similar Twitter users and even represent short text sentences using Word2Vec [17][18].

We applied both existing and proposed methods to generate k user clusters using both the views. We evaluated consensus using Cohen's kappa between the views. We report the results obtained in the fifth iteration for both the views where the consensus between the existing approach is 0.053, the proposed approach is 0.608. Since existing approaches are not able to generate homogeneous clustering results from both views, we are not able to generate diverse and non-diverse crowds. We therefore carried out the WoC experiment only with our proposed method instead.

We used the k user clusters generated by the proposed method to form the diverse crowd and non-diverse crowds by selecting one user from each cluster to form a diverse crowd of k users and selecting k users from one of the clusters to form a non-diverse crowd. Using this approach, we then generate 5000 random unique groups of different sizes, for each category (diverse and non-diverse).

We evaluate the crowd wisdom by calculating wisdom score of each group of size n, G = (C₁, C₂, ..., Cₙ) over 25 weeks, with group size ranging 8 to 18. The wisdom score of each group can be computed by \[ \frac{\sum_{i=1}^{25} \text{Mod}(C_i)}{25} \], where \( \text{Mod}(C_i) \) represents the score of the individual captain receiving the most “votes” from the group in the \( i^{th} \) game week. We observed the following when we compared the results of diverse crowds and non-diverse crowds in
player's view, team view and both the views together. Figure 14 and 15 show the captain score for diverse and non-diverse crowd of player’s and team view respectively.

Figure 14: Captain scores for diverse and non-diverse crowd vs. crowd size for player’s view
Figure 15: Captain scores for diverse and non-diverse crowd vs. crowd size for team view

Figure 16: Captain scores for diverse and non-diverse crowd vs. group size for both the views

The above results (Figure 16th) show that diverse user groups from multi-view clustering continually give higher average wisdom scores.
6. Related Work

The motivation for our method is to cluster the data points that share the similar attributes in the multiple views by achieving the maximum clustering agreement between the views with the efficient and scalable approach.

In this section, we discuss some of the existing work in multi-view clustering. Several multi-view clustering algorithms have been proposed in the past which deal with multi-view data. Most of these algorithms extract shared attributes from multiple views and apply simple clustering algorithms like k-means on the extracted attributes. Canonical Correlation Analysis (CCA) [19] is one best example of this kind. Some other algorithms utilize the data in multiple views as part of the clustering algorithm. Co-EM is one best example of this kind. Co-EM [20] iteratively computes the clustering results of each view and uses them to update the other views for a certain number of iterations.

Co-training with multi-view clustering was first introduced by Blum and Mitchell [11]. The idea of co-training is to train various views to maximize the mutual agreement iteratively. Spectral clustering [21] was first used in multi-view clustering by de Sa in 2005 by constructing a bipartite graph of two views [22]. In 2007, Zhou & Burges developed a framework which is good for multiple graphs but not the best for a single graph [12]. Kumar and Daume have also done extensive work in this area [6]. They applied the idea of co-training using spectral clustering for multi-view data where the similarity matrix of
one view will be computed by the eigenvector of the Laplacian in another view and vice versa. This process is carried out through a certain number of iterations. This algorithm often does not converge.
7. Conclusion

We introduce a novel framework which facilitates measuring crowd diversity and technique to understand whether and how diversity is affecting the collective intelligence. Here we examined whether it is possible to extract measures of a crowd’s diversity based on its members’ social media (Twitter) communications and whether such measures can lead to a selection of wiser crowds. We describe an efficient and scalable power iteration based co-training approach to achieve convergence for multi-view clustering. Experiments on real, synthetic datasets show that our algorithm can converge faster than existing algorithms. Experimental results on WoC datasets also show that diverse crowd outperforms non-diverse crowd.
Bibliography


Appendix A

This appendix lists the snippet of code used in our work. This snippet of code contains the Class for co-training with power iteration clustering.

```java
public class powerIteration_Cotraining {

    public static void Co_training(List<double[][]> data, int num_views,
                                    int numClust, double[] sigma, int[] truth, double projev, int numiter, int val)
            throws Exception {
        SimilarityMeasure dist = new SimilarityMeasure();

        List<double[][]> K = new ArrayList<double[][]>();
        List<double[][]> V = new ArrayList<double[][]>();
        List<int[]> C = new ArrayList<int[]>();

        int N = data.get(0).length;

        for (int i = 0; i < num_views; i++) {
            System.out.println("Computing kernel for view " + (i + 1));
            K.add(dist.constructKernel(data.get(i), data.get(i), sigma[i]));

            ArrayList basRes = new ArrayList<>();
            basRes = baseline_poweriteration_method(K.get(i), numClust, truth, projev);

            System.out.println(" nmi " + (double) basRes.get(2));
            V.add((double[][]) basRes.get(1));
            C.add((int[]) basRes.get(0));
        }

        List<double[][]> X = new ArrayList<double[][]>();
        List<double[][]> Y = new ArrayList<double[][]>();
        List<double[][]> Y_norm = new ArrayList<double[][]>();

        X = V;
        Y = K;
        Y_norm = Y;

        double[][] Sall = new double[K.get(0).length][K.get(0).length];

        System.out.println("Starting Co-training approach");
        for (int i = 0; i < numiter; i++) {
            System.out.println("\nIteration ... " + (i + 1));
            double[][] totalClustResult = new double[N][N];
            ArrayList<double[][]> ClustResult = new ArrayList<double[][]>();

            for (int j = 0; j < num_views; j++) {
                int[] clust = C.get(j);
            }
        }
    }
}
```
double[][] Kval = K.get(j);
double[][] tmpClustResult = new double[N][N];
for(int row_iter=0; row_iter<N-1; row_iter++)
{
    tmpClustResult[row_iter][row_iter] = 1;
    for (int col_iter = row_iter+1; col_iter<N; col_iter++)
    {
        if(Kval[row_iter][col_iter] != 0)
        {
            if(clust[row_iter] == clust[col_iter])
            {
                tmpClustResult[row_iter][col_iter] = 1;
                tmpClustResult[row_iter][col_iter] = 1;
            }
            else
            {
                tmpClustResult[row_iter][col_iter] = 0;
                tmpClustResult[row_iter][col_iter] = 0;
            }
        }
        else
        {
            tmpClustResult[row_iter][col_iter] = 0;
            tmpClustResult[row_iter][col_iter] = 0;
        }
    }
}
ClustResult.add(j,tmpClustResult);
totalClustResult = MatrixOperations.sumMatrix(totalClustResult,tmpClustResult);
}
double alpha = (double)1/num_views;
for (int j = 0; j < num_views; j++) {
    double[][] Xtmp = X.get(j);
    double[][] Ktmp = K.get(j);
    Matrix Kmat = new Matrix(Ktmp);
    Matrix totalClustResultMat = new Matrix(totalClustResult);
    Matrix ClustResultMat = new Matrix(ClustResult.get(j));
    Matrix YtmpMat = totalClustResultMat.minus(ClustResultMat);
    Matrix Ymat = Kmat.plus(YtmpMat.times(alpha));
    Ymat = Ymat.plus(Ymat.transpose());
    double[][] YnTmp = Ymat.getArray();
    for (int ll = 0; ll < YnTmp.length; ll++) {
        for (int lm = 0; lm < YnTmp.length; lm++) {
            YnTmp[ll][lm] = YnTmp[ll][lm] / 2;
        }
    }
    Y.set(j, YnTmp);
    Y_norm.set(j, YnTmp);
}
ArrayList basRes = new ArrayList<>();
basRes = baseline_poweriteration_method(YnTmp, numClust, truth, projev);
X.set(j, (double[][]) basRes.get(1));
C.set(j, (int[]) basRes.get(0));

System.out.println(" nmi "+(double) basRes.get(2));
}