Multi-view Clustering of Social-based Data

Iain J. Cruickshank

CMU-ISR-20-109
July 2020

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Kathleen M. Carley (Chair)
L. Rick Carley
J. Zico Kolter
Tanya Berger-Wolf (The Ohio State University)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Societal Computing.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship (DGE 1745016), Department of Defense Minerva Initiative (N00014-15-1-2797), and Office of Naval Research Multidisciplinary University Research Initiative (N00014-17-1-2675). Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation, Department of Defense, or the Office of Naval Research.

Keywords: machine learning, computational social science, clustering, social network analysis, graph learning, community detection, multi-view data, multi-modal data, cluster ensembling, malware analysis

This work is dedicated to my wonderful family; to my children for teaching me how to be curious, to my father for inspiring me, and, most especially, to my partner and wife, Jenn, who believed in me when no one else — to include myself — did.

Abstract

Real-world, social phenomena produce various types of data, like explicit networks or user-emitted text. When different sets of data describe the same entities, the data is termed multi-view or multi-modal. A distinct advantage of multi-view data is that different views may better capture different aspects of the latent structure of the data. However, there are difficulties in combining that data to produce something like a clustering of the data. Multi-view clustering techniques, primarily developed for image or biological use cases or network only use cases, have typically not been used for clustering social-based use cases. I investigate the use of multi-view clustering on various social-based, multi-view data sets, and propose new techniques for multi-view clustering of social-based data.

In the first part of this thesis I discuss the use of multi-view clustering for social-based data, and propose a new paradigm and new techniques for multi-view clustering. In chapter two I propose a new hybrid paradigm of multi-view clustering, which combines elements of late paradigm and intermediate paradigm integration. I test the various intermediate, late, and hybrid paradigm algorithms on a wide range of benchmark data sets from social-based data scenarios. The results of the empirical testing demonstrate a wide variance in the performance of multi-view clustering techniques. This is in part because social-based data often have high inter- and intra-view variances that are not present in other data scenarios, which presents difficulties for existing techniques. Only two techniques proposed in the chapter have good performance across all of the data sets and are robust to inter- and intra-view differences. From the results in chapter two, I devise a new algorithm based in network modularity and graph learning to cluster multi-view social data in chapter three. I present the results of a series of empirical tests of the new technique, as well as possible variations on the technique. The results demonstrate that the presented technique often performs well across a wide range of social-scenarios that give rise to multi-view data, is scalable to large data sets, and is robust to inter- and intra-view variance.

In the second part of this thesis I use the new techniques to do clustering analyses of real-world data. In chapter four I use multi-view clustering on Twitter data collected during the initial stages of the COVID-19 pandemic. This analysis is the first ever use of multi-view clustering to cluster hashtags from large, social-media data sets. The results display that hashtags form topical clusters and that these topical clusters have changed over the course of the pandemic. In chapter five I use multi-view clustering to cluster malware samples. The results demonstrate that a multi-view clustering of malware samples provides insight into communities of malware use, and confirm the techniques developed can be applied to a wide range of social-based data scenarios.

In sum, I demonstrate the suitability of, and create techniques for, multi-view clustering of complex, multi-view, social-based data. This thesis advances practical clustering analyses of large-scale, noisy, social-based data and contributes to the field of multi-view clustering in general.

Acknowledgments

The Ph.D. experience has been one of the most rewarding of my life. It was not an easy journey and I don't think I have ever failed so many times, in so many ways, than I did while doing a Ph.D. As such, I have grown more than I thought possible, both in terms of my subject matter expertise, but also as a person in my outlook on problems and what really matters in life. This opportunity was a rare time in life where I could think without limits and listen to some of the greatest ideas I have ever been exposed to. If given the opportunity to go back in time, I would absolutely do this again — except maybe with trying even more failing ideas.

I would first like to acknowledge the National Science Foundation for funding my Ph.D. studies. As a junior officer in the United States Army, an opportunity like studying for a Ph.D. would normally be denied to me. So, I would like to thank the National Science Foundation, and those good people who saw fit to fund me, despite my juniority.

There are many individuals without whom I would not have been able to do this Ph.D. First, I would like to thank Dr. Kathleen M. Carley, my advisor, for investing in me and mentoring me through this process. Ever since I was a Cadet, I have been impressed with Dr. Carley's willingness to mentor and listen to ideas (no matter how bad they might be). Second, I would also like to thank my committee, Dr. Rick Carley, Dr. Zico Kolter, and Dr. Tanya Berger-Wolf for all of their help. Despite my many missteps in this process, all of them have taken time to mentor me personally and keep me on the path to completion. Third, I would like to thank all of my colleagues, who have been instrumental in any success that I have had. I would like to thank my cohort peers David Beskow and Tom Magelinski for helping get through all of the course work we did together, Sumeet Kumar, Matt Babcock, Geoff Dobson, and Daniel Smullen for teaching me how to be a successful Ph.D. student. And, I would like to thank Gian Maria Campedelli for helping me become an actual researcher and Mihovil Bartulovic for all the work he did as my fellow teaching assistant.

I would also like to acknowledge the huge debt of gratitude that I owe to those who have mentored me over the years. This success is really their success. I would like to thank Dr. Ted Hromadka who first inspired to study mathematics and continues to inspire me to be a better scholar and leader. I would also like to thank Drs. Ed and Jenny Fusilier and Dr. Amanda Beecher who showed me the beauty of mathematics and who have continually encouraged me to keep pushing my education forward. I would never have come this far without their mentorship and wisdom.

Finally, and most importantly, I would like to thank my family. Their unyielding support and love has made me who I am.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction and Background

1.1 Introduction

In today's social media-laden, digital world, researchers are increasingly able to see many different *views* of social interactions. Everything from social media networks and content to observed online and in-person interactions can all be recorded and used to understand important structures within those social actors like communities or patterns of usage. Despite the benefit provided by all of these different views of the data, this complexity can also pose challenges for computationally analyzing the data. One particular challenge is how to cluster multi-view, social-based data — which could contain any number and variety of data types, like network and non-network data — in order to do things like explore meso-structures in the data, find communities, or characterize interesting patterns of usage. Typically, only one of these different views of the data are used in exploring structures, like clusters, in the data due to difficulty with using multiple different types of data in one cohesive clustering. So, to tackle this problem, I propose the use of multi-view clustering; in this thesis investigate how multi-view clustering can be applied to social-based data scenarios.

The field of multi-view clustering, like many other multi-view, or multi-modal, machine learning fields rests on the idea that not only more data, but incorporation of different types of data, can lead to better outcomes. In the case of multi-view clustering, many techniques have been developed around multi-view data of images or genetic interactions. Additionally, within the discipline of network science there have been a plethora of techniques developed to analyze individuals that interact in many different networks. Generally, techniques which can incorporate additional modes, or views, of the data perform better at tasks like clustering than those techniques which cannot. So, as social-based scenarios, especially from social media sites, are giving rise to more types of data, it is important to investigate how these multi-view and multi-modal techniques can be used with social-based, multi-view data scenarios. In particular, it is important to investigate techniques which are capable of taking any kind of data that can arise from social interactions, like networks and text or user attributes.

In this thesis, I will investigate the use of multi-view clustering on social-based data. In particular, I will investigate the use of multi-view clustering of social-based data scenarios that can give rise to multiple types data in the different views of the data, and not just those techniques

which can only handle one type of view (i.e. just network views or non-network views). In contrast to previous work, this work is the first to propose a multi-view data paradigm to cluster social-based data. It also is the first work to empirically investigate the use of multi-view clustering techniques on social based data; most previous work on multi-view clustering techniques focuses on image or biological data. As a result of this work, a new multi-view clustering technique has been developed which not only shows strong empirical performance on benchmark data sets, but also great utility in investigation of clusters in real-world, multi-view data.

The thesis is organized as follows: In the first section, I give some background on multi-view clustering and its various applications and paradigms of techniques. I will also define some of the key concepts and terms used throughout the thesis. In the second chapter, I will investigate various techniques from the different paradigms of multi-view clustering in terms of their performance in clustering benchmark, multi-view, social-based data from several different social-based data scenarios. I also propose a new hybrid paradigm for clustering social-based data and some techniques from this hybrid paradigm. In the third chapter, I develop a new technique based on lessons from the previous chapter for clustering multi-view, social-based data and investigate its performance and the performance of variations in the technique. In the final two chapters, I use the previously developed techniques on actual data from real-world problems to gain meaningful insight into those problems. In the fourth chapter, I use multi-view clustering to investigate hashtag usage on Twitter during the COVID-19 pandemic. And, in the fifth chapter, I use multi-view clustering to understand communities of malware samples.

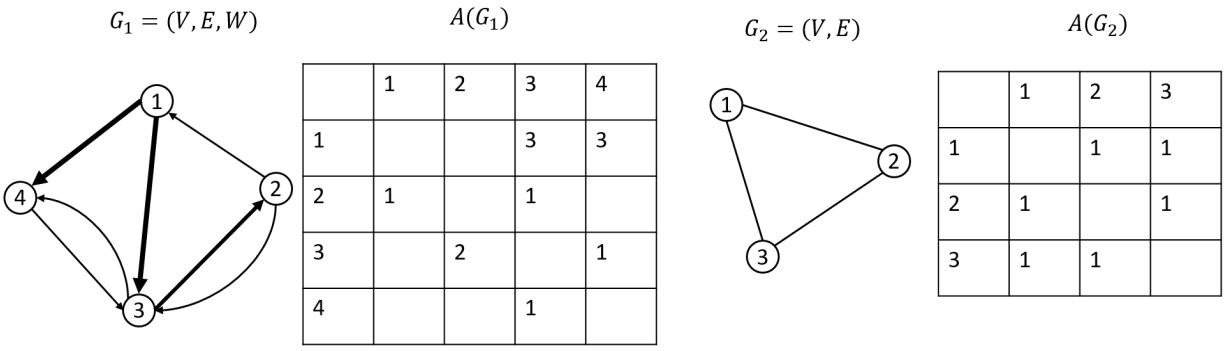
1.2 Background

This section summarizes the key concepts and frequently used terms in the thesis as well as covering the relevant background research. This section provides all of the meaningful context for the reader to understand the remaining chapters of the thesis and their research value.

1.2.1 Definitions

This section details some of the key concepts and terms that will be used throughout the thesis.

Graph The first key concept used throughout this thesis is a *graph*. A graph, G, is a set of homogeneous objects called vertices, V that have relationships between each other. These relationships are called edges, denoted as E. A graph can further have a weight, W, on each of the edges which represents the strength of the relationship between the vertices at the end points of the edge. Finally, an edge may have a direction which indicates a one-way relationship or flow. Directed edges can also be referred to as arcs [94]. The connections of a graph may also be summarized as an adjacency matrix, A, which is an $|V|$ by $|V|$ matrix that has entries representing the edges (and their weights in the case of a weighted graph). The following figure 1.1 displays some of these different definitions of a graph:



	1	2	3	4
1			3	3
2	1		1	
3		2		1
4			1	

	1	2	3
1		1	1
2	1		1
3	1	1	

Figure 1.1: On the left is a directed, weighted graph, where arrows indicate the direction of the edge and the width of the edge indicates the weight, with its associated adjacency matrix. On the right is a undirected, unweighted graph with its associated adjacency matrix

An unweighted, directed or undirected graph, G, will be denoted as $G = (V, E)$ and a weighted undirected or directed graph as $G = (V, E, W)$. It should be noted that the term 'network' is often synonymous with the term graph and that its constituent terms of 'node' and 'link' correspond directly to 'vertex' and 'edge' respectively [94]. The two frameworks will be used interchangeably in this thesis, and use of one or the other is meant to be consistent with the research that gave rise to findings or methods being mentioned. Typically network will be used for actual observed relations between actors, while a graph will be used to refer to inferred relations or calculated similarities between entities.

From the definition of the graph we can then derive another important definition, that of the neighbors of a vertex. The neighbors of some vertex, u, are all those vertices of V which u has an edge to them, $e_{u,v} \in E, \forall v \neq u$. This is equivalent to the nonzero elements of the row of the adjacency matrix assigned to vertex u. The neighbors of any given vertex, u, are denoted as N_u.

Clusters and Clustering In this work, *clustering* is the task of splitting a collection of objects into non-overlapping subsets, such that all objects are assigned to exactly on subset, and each subsets' objects are more similar to each other than they are to any other subsets' objects [14], [7]. So, at the end of performing a clustering on a data set, a labeling is produced where each object in the data set now has a label for the cluster that that object belongs to. In this work, a *cluster* is a particular label or subset of the data, and a *clustering* is the collection of all of the labels or subsets produced by a single application of a clustering algorithm to the data.

Multi-view or Multi-modal Data Another concept that is key to understanding this proposal is multi-view, or multi-modal, data. In this work, multi-view data is a data set which has different sub-divisions of the data (which could be of different types of data) each of which describe the *same* set of entities [31], [58], [141], [12]. Furthermore, each entity *can be* described in all of the views of the data, but may not always have data within each view [12], [32]. By these definitions, multi-view data has features which are inherently correlated; each view of the data should be correlated in some respect with every other view since they are describing the same phenomena. An example of multi-view data would be news about an event; news articles often contain text and images which both are data describing the same event [12]. Another example would be

an image of something which has been put through many different filters. Each version of the image could be a view of the data of the image [12]. In fact, this latter example represents much of the types of multi-view data that disciplines like graph learning have been applied to [10], [9], [106]. Overall, multi-view data is data which either explicitly has different views describing the same set of entities or data which can be made multi-view through different representations of, or measurements on, the same data.

Common Notations Throughout this thesis, I will use the common mathematical notational rules to include having matrices be represented by uppercase letters, vectors or constants represented by lowercase letter, and parameters by Greek symbols. In addition the following table, Table 1.2.1, summarizes some notations that are particular to this work, for reference by the reader.

Commonly used Symbols and Definitions	
n	the number of objects in the data set
m	the number of views in the data set
k	the number of clusters of a data set
d/d^v	the dimensionality of a set of features/ set of features within a particular view, v
c_i	a cluster produced by a clustering technique
X/X^v	the matrix of features vectors/ matrix of features vectors for a particular view, v
G/G^v	a network or graph/ the network or graph for a particular view, v
V	the set of all vertices or nodes in a graph or network
E	the set of all edges or links in a graph or network
A/A^v	the adjacency matrix of a particular graph or network/ adjacency matrix for a particular view, v
P/P^v	the clusters or partitions produced by a clustering algorithm/ clusters for a particular view, v

1.2.2 Related and Foundational Research

In this section I will review the fields of study and previous research that is fundamental to the contributions in this work. I will begin with detailing the use of multi-view data in machine learning. I will then describe the specific work on unsupervised learning with multi-view data. As this work focuses on clustering with multi-view, social-based data I will also then detail the research in the field of social network analysis that attempts to address finding communities in multi-view, social-based data. In particular, I will outline the research done in clustering attributed graphs and clustering of multiplex and multi-layer social networks.

Machine Learning with multi-view Data

Multi-view data has been used for many types of machine learning tasks. A recent survey of the works in machine learning with multi-view data by Baltrušaitis et al. generally categorizes the challenges of using multi-view data as representation, translation, alignment, fusion, and co-learning [11], [12]. In this work we are primarily investigating the challenge of fusion of

multiple views of data in order to produce meaningful clusters of the data [12]. Fusing various types of multi-view data remains an active area of research due to its salience in fields like medicine and computer vision among others [160], [12], [99]. Within this framework of data fusion, the techniques of fusing multi-view data for an unsupervised learning task break into three paradigms: early integration, intermediate integration, and late integration [12], [160]. Early integration approaches, which historically have made up the bulk of all multi-view data uses typically feature concatenation of the different views into one view [12]. In essence, early integration techniques create one feature vector from all of the view feature vectors and then apply a particular machine learning algorithm to this new feature set. The strength of early integration approaches are their simplicity since they can be used without having to design any kind of specialized machine learning algorithms. The weakness of early integration approaches are that concatenating the information before using it in a machine learning algorithm can cause certain view-specific patterns to be masked by the other views' features [12]. Additionally, some types of data simply do not concatenate easily, such as network and non-network data. Figure 1.2 displays an example of an early integration approach.

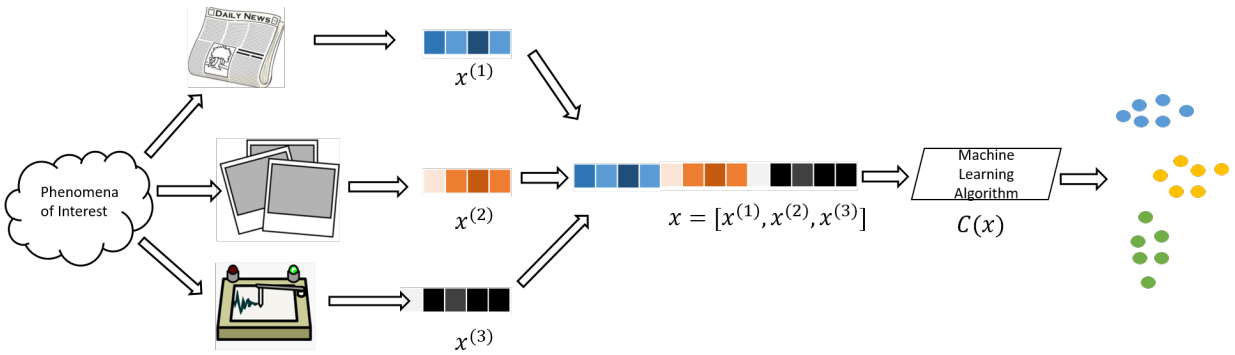


Figure 1.2: Schematic of the early integration paradigm. An phenomenon of interest can give rise to many different types of data. In the early integration paradigm, the different types of data are converted to the same type of features, and then these features are just concatenated together and used in a standard machine learning algorithm.

Late integration approaches work by applying a machine learning algorithm to each view independently and then averaging or otherwise ensembling together the output of each view's machine learning algorithms. Late integration techniques derive a new feature vector of outputs from each view's machine learning algorithms and then use that feature vector in a final machine learning algorithm. The main advantage of late integration approaches is that they can leverage specialty algorithms designed for certain types of data (i.e. using network clustering algorithms or algorithms designed specifically to cluster text for the network and text views of the data) and can be parallelized in execution across the views. The main drawback of late integration approaches is that each view is treated independent of the other views which means complementary information between the views is not used and that just using the outputs of each view's machine learning algorithms will entail some loss of information which often degrades the performance of the algorithms. Figure 1.3 displays an example of a late integration approach.

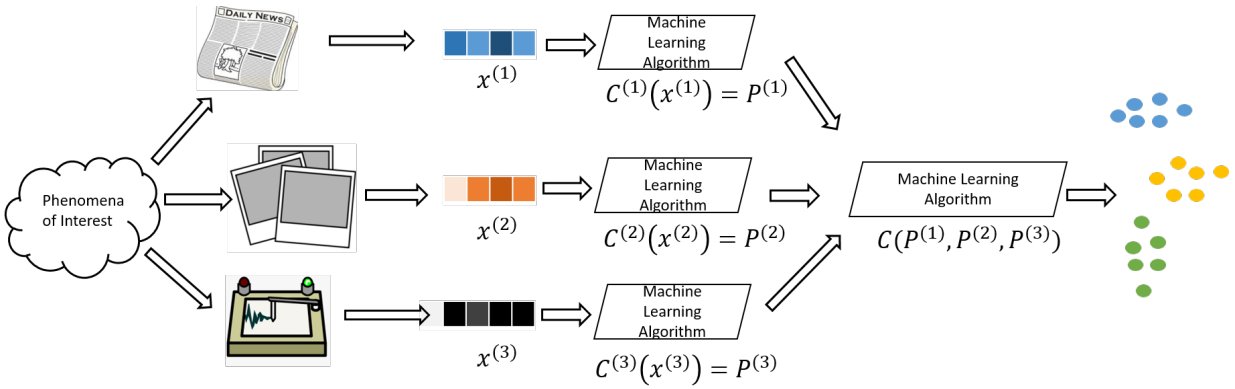


Figure 1.3: Schematic of the late integration paradigm. An phenomenon of interest can give rise to many different types of data. In the late integration paradigm, the different types of data are clustered or otherwise transformed by whatever algorithm makes sense for that type of data. The resulting view labels are then fed to a ensembling technique, which is generally another machine learning algorithm, to produce the final labels.

Intermediate integration techniques are those which work directly on the multi-view data in a multi-view format to produce the desired output. Typically each view's features are cast to some kind of compatible data format, like graphs, and then a loss function that combines the loss across each of the views is used in the machine learning algorithm. The main advantage of intermediate integration approaches is that they take advantage of information present within and between the views of the data. Their primary disadvantage is that they require all of the views of the data to be of a certain type (i.e. all graphs) which can limit the types of multi-view data that they can be used on. Furthermore, different data set ups can require completely new loss functions, which makes many of the techniques limited in their application domains. Figure 1.4 displays an example of an intermediate integration approach.

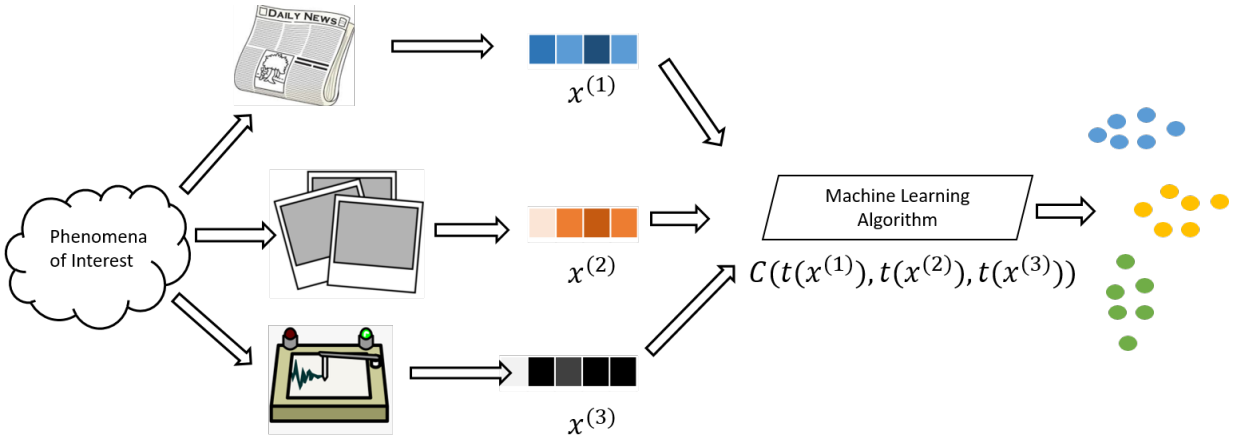


Figure 1.4: Schematic of the intermediate integration paradigm. An phenomenon of interest can give rise to many different types of data. These different types of data are then featurized and a machine learning algorithm which has a loss function that works on all of the views simultaneously to produce the desired output. Often, each of the views are mapped to the same space or data type (i.e. a graph) as part of the algorithm.

While each of the integration strategies have their strengths and weaknesses, late or intermediate integration approaches are generally seen as superior approaches for dealing with

multi-view data [12], [160], [123]. This is due to these approaches better using aspects of the multi-view data like complementary information between views or algorithms that are specific to certain types of data within the views. Overall, while the use of multi-view data can lead to improved results for machine learning, it also presents distinct challenges for fusing the data to be used for a machine learning task.

To date most of the data being characterized as multi-view data — and, by corollary, driving algorithmic development for multi-view data — has been limited to a few domains. The domains that almost exclusively dominate multi-view data usage in machine learning are audio-visual-text, image, and medical/genetic. Perhaps the oldest use of multi-view data has been for solving problems in the combined domains of audio-visual and text [12]. Within this domain of data, problems like providing text annotations for videos, recognizing speech, and translating text between languages form many of the multi-view problems [12], [97], [118], [79], [46]. Many of these problem setups and corresponding algorithms exploit labeled data (or, partially labeled data), and deep neural network architectures.

More recently, image data has become increasingly characterized as multi-view data [10], [157], [1]. Many of the approaches designed for image data utilize different filters, representations, or similarity measurements on the same set of images and represent each of these differences as a different view of the image. Additionally, many of the techniques used on multi-view image data exploit graph representations of the data and properties of graphs for fusing the data [10], [9], [157], [43], [134]. Much like the techniques used in the image domain, many of the most successful multi-view techniques used for biological and genetic data also typically use graph-based representations [107]. Some techniques utilize diffusion processes in order to fuse or otherwise enhance graph-based biological structures like protein-protein interaction networks [135], [136]. Other techniques use ensembles of graphs or label information to otherwise aid in the fusion of the graphs of the different views of the biological data [160], [56], [99], [91], [100]. Taken together, audio-visual, biological/genetic, and image data make up nearly all of the data domains that have exploited multi-view data characterizations for machine learning.

Most recently, a few recent studies have begun using social data in a multi-view way for supervised learning problems. In this context, social data is that data which is generated by a social process, like data coming from online social networks or social interactions in an organization. Some recent works using multi-view, online social data have found that using more modalities of online social media data than just the text emitted by users or their social interactions have led to gains in supervised learning. In particular, recent research has found the fusion of content data (i.e. text) with network diffusion data (i.e. to whom that text spread to in an online social network) to be especially useful for detecting fake news and characterizing online conversations [104], [116], [117], [77]. Thus, solutions for some very tricky supervised learning problems in a social data domain have received substantial boosts in predictive performance by the inclusion of multi-view data into their algorithms.

Clustering of multi-view Data

In this section I will describe the primary techniques used in multi-view clustering. As was detailed in the previous section, multi-view clustering techniques break into the three paradigms of early, intermediate, and late integration techniques. Since early integration techniques use

standard clustering algorithms, after transforming the features from the different views into one feature, this review will focus on intermediate and late integration clustering techniques.

The advantage of multi-view clustering derives form two basic principles: *complementarity* and *consensus*. The complementarity principle is that multiple views of data can be employed to more comprehensively and accurately describe the phenomena of interest [147]. With multi-view data, any single view is sufficient for the clustering task, but the combination of multiple views can provide complementary information to any single view which improves the clustering task. The consensus principle is the idea that a clustering solution should be maximally consistent across the multiple views [147]. This principle implies that finding a clustering solution that reduces the error between it and all of the view clustering solutions will be the optimal clustering solution. So, any successful multi-view clustering algorithm should be able to take into account complementary information between the views of the data and seek a final clustering which is maximally consistent with the clusterings from each view.

Intermediate Integration Intermediate integration, multi-view clustering methods are those methods which attempt to cluster features from all of the views simultaneously. Generally speaking, these methods require that each view of the data be projected into the same space and have a loss function that is defined for all of the views concurrently. Several techniques that were designed for uni-modal data have been extended for use on multi-view data sets [31], [58], [141], [1]. Typically, these methods modify the loss function for a uni-modal clustering technique, such as k-Means, to have the computation of loss be across all of the views of the data [148]. These techniques often employ multiple kernel learning where there is a different pre-defined kernel for each of the views which allows for all of the views to be better accounted for in a singular loss function [147].

More recently, intermediate integration techniques have focused on using graphs as the common data format for multi-view clustering and the use of factorization procedures to produce a common subspace due to their high performance on empirical tests in the image and genetic data domains [147], [107]. In the case of graph-based intermediate integration techniques there are several techniques that use a spectral clustering framework [58], [141], [150], [143], [153], [140]. Some of the spectral clustering-based methods additionally use techniques which automatically learn weights for the different graphs from the different views to allow for discounting less useful views of the data[150], [153]. Other techniques in the graph-based methods use Tensor Factorization techniques or a form of Non-Negative Matrix Factorization to produce latent features which can be clustered by a technique like k-Means [153], [60], [139], [55], [151]. In this way, these graph-based techniques employing matrix factorization can often closely resemble those techniques that attempt to just find a common subspace across views [147]. The primary difference is typically in the loss functions which can either take a graph as input or a raw feature matrix as input [147]. Many of these factorization-based techniques also include additional regularization terms in the loss function to encourage sparse or low-rank results, which can be used to produce better common subspaces or graphs for subsequent clustering [147], [139], [151]. Additionally, some factorization-based techniques also include a model of the view data that has an explicit term for view inconsistencies or error [147], [143], [82], [139]. Inclusion of such a view inconsistency term allows for a more resilient clustering solution since differences between

the views of the data have less of an impact on the joint subspace or graph.

The primary strength of intermediate integration techniques is their ability to exploit the complementary principle when producing clustering solutions. As such, they are often considered the superior class of techniques for clustering multi-view data [123], [160]. However, these methods rely on data being the same format or the same space, which can present challenges for heterogeneous data formats like networks and non-network data (i.e text) [147]. Furthermore, many techniques in intermediate integration can also be computationally demanding and have certain initialization and hyper-parameter dependencies that can substantially affect their performance.

Late Integration Late integration, multi-view clustering methods are those methods which cluster each view of the data independently and then attempt to find a consensus clustering across all of the view clusterings. The vast majority of the late integration methods are cluster ensembling methods. Cluster ensembling is the idea that multiple clusterings are taken of the same set of data, and then the partitions are combined to produce a better partition of the data set by a consensus method [18]. In this way, cluster ensembling techniques are targeting the consensus principle involved in clustering of multi-view data. In the original problem setup for cluster ensembling the consensus method only has access to the the individual partitions and no other information about the underlying data [119], [18]. While cluster ensembling was originally designed for uni-modal clustering as a means of providing for more robust and accurate clusters, the methodology has been recently extended to explicitly handle multi-view data scenarios [18], [152].

The main methodological traditions of cluster ensembling are optimization-based and graph-based. In optimization-based cluster ensembling, the consensus function typically consists of a loss function and an optimization procedure for the loss function [18], [152]. In particular, most optimization-based methods have a loss function that measures the difference between the consensus clustering and each of the ensemble measure with an optimization procedure to reduce this difference.

Since the optimization-based methods can suffer from having NP-hard optimization schemes and scalability to larger data sets, graph-based methods have been the preferred methods for cluster ensembling [18]. Within graph-based methods there are several proposed techniques for constructing the graph, producing weightings for the graph, and clustering the graph for the final consensus partition [18], [152]. For constructing a graph of all the view partitions three main methods are used. The first method is to produce a co-association graph of the objects being clustered by placing edges between the objects in the graph if the share clusters in the view clusterings [119], [18], [152], [71], [65], [64], [85]. The second main method of constructing a graph from the view partitions is to create a meta-cluster graph where the vertices are now the view clusters and the edges represent some measure of how many objects the view clusters share [119], [64], [65]. The third main method of constructing a graph from the view partitions is to create an object by view cluster bipartite graph where an edge represents that an object belongs to a view cluster [65], [47]. With these basic graph formats, there are many proposed of ways of weighting the graphs in order to better emphasize good view clusterings over bad view clusterings [152]. Typically, the weighting of an edge in any of these methods encompasses the strength of the connection between its endpoints in the view clusters and the consistency of the

view clusters that indicate the endpoints should have an edge, with all of the other view clusters [85], [152], [64], [6], [65], [71]. Once a graph (possibly weighted) has been constructed from the view partitions there have also been several means of clustering this graph to produce the final partitions. The most often used means of the final clustering are Spectral, METIS, and agglomerative hierarchical clustering [119], [18], [152].

Most recently, some authors have sought methods that combine the final clustering with construction of the graphs from the view partitions. In particular, several recent works have used a spectral clustering paradigm combined with a graph learning paradigm for cluster ensembling [125], [124], [156], [86], [75], [126]. These methods typically combine an iterative optimization procedure that first minimizes the final clustering error and then uses that solution to produce optimal graphs from the view clusters, and so on.

The primary strength of late integration techniques is their ability to exploit the consensus principle and view-specific clustering techniques. They are, however, generally regarded as inferior to intermediate integration techniques as they inherently do not take into account the complementary principle by clustering each view independently. This view of late integration techniques has been empirically challenged by some recent works, and so for any given data scenario it is not clear that an intermediate integration technique will perform better than a late integration technique [124], [147]. Thus, the late integration paradigm is a valid way of designing methods for multi-view clustering.

Community Detection in multi-view Social Networks

Within the realm of social-based data, and in particular social network analysis, there are two main areas of research that deal with multi-view data. The first is multi-layer or multiplex social network analysis and the second is attributed network clustering. Multiplex networks are networks in which a node has more than one type of link connecting it to other nodes [3]. These networks can be modeled as a collection of networks that are defined over the same nodeset but have different links within each of the networks. So, in this data format, each network represents a possible view of the data. Multi-layer networks contain the same networks as multiplex networks, but with the addition of inter-layer links where a node in one layer can be connected to nodes in other layers [3]. As with multiplex networks, each layer can be considered a view of the data. Attributed networks are networks which also have additional information on the nodes [32]. So, an attributed network will have two views of data; one view which is the network itself and a second view of features describing the nodes present in the network. Thus, the social-based data models of multiplex and multi-layer networks and attributed networks present limited versions of multi-view data, which can be clustered for tasks like community detection.

Multi-layer and Multiplex Network Clustering Over the years, several methods have been put forward to find clusters in multiplex and multi-layer methods. The most common way of finding communities within these networks is to compress all of the layers of the network into one network (typically by directly adding the graphs together) and use a more traditional network clustering technique [3], [41]. However, since compressing all of the layers, or views, of the data into one entails a loss of information, some methods have been put forward to correct for this loss of information when combining the layers, or otherwise determine what layers can be combined

without losing too much information [53], [42]. In this way, the compression of multi-layer and multiplex networks represents the early integration approach to clustering this class of multi-view data.

There have also been numerous intermediate integration approaches proposed for multi-layer and multiplex network clustering. Generally, the major difference between the various intermediate integration clustering approaches is the loss function that they are minimizing. The most widely used loss function is the Network Modularity function which was extended to cover both multi-layer and multiplex networks [93], [101]. Other loss functions include network flows and clustering coefficients which can be optimized by the same procedures as was done for Network Modularity [20], [37]. Another common intermediate integration approach to clustering multi-layer and multiplex networks are stochastic block models which are generative models that assume the community structure present in the data arises from statistical properties of the links [98], [101], [68]. Finally, some other methodological ideas like spectral clustering have also been adopted to multi-layer and multiplex network clustering [39].

Only a few, recent works exist in the late integration paradigm for clustering multiplex or multi-layer networks. Generally speaking, all of these works use a network-based clustering technique (i.e. Louvain [15]) on each of the views independently, and then create a co-association graph (i.e. each object has a link to another object weighted by the number of view clusters that those two objects share) and cluster that graph to get the final clustering [81], [121], [90], [122]. In contrast to most co-association late integration techniques, these methods typically feature a sparsification step on the co-association graph (as it is often dense), to make it more amenable to further network clustering, and apply network clustering and co-association graph construction in an iterative fashion to refine the final clusterings [81], [121], [90], [122]. Of note, these techniques have been applied to social media data with good results for community detection [59]. Also, there is one recent late integration multiplex network clustering method that included an additional step in the refinement iterations to add back in edges to the co-association graph based on the presence of that edge in all of the networks from the original, multiplex network [121], [90]. In this way, a limited amount of intermediate integration information is infused into the late integration model to produce better results. Overall, while there have been some methods from the late integration paradigm applied to multiplex network clustering, these methods are largely overshadowed by the intermediate network clustering techniques in actual usage.

Attributed Network Clustering Attributed network clustering has several methods from all three of the multi-view clustering paradigms. In the early integration paradigm most methods tend to follow the pattern of creating a new graph from the given network and the nodal attributes and then clustering this new graph to produce the final clusters. Most methods of early integration adopt a combination strategy of directly combining the aspects from the network view and aspects of the attribute view in a convex combination, $W(e_{ij}, \alpha) = \alpha W_T(e_{ij}) + (1 - \alpha)W_s(e_{ij})$, where W_T and W_S are the functions for the network views and the attributes respectively [32], [102], [4], [103]. This final combined graph is then clustered by standard network clustering techniques to produce the final clusters. A salient variation on this model is to include adding in the attributes as nodes themselves to the given network to produce the final network [30]. Additionally, recent research has also tied the network model construction to the clustering output to

give better combined models of the network and attributes views [22], [21].

There have also been a number of intermediate and late integration techniques for clustering attributed graphs. Within the paradigm of intermediate integration techniques, most methods tend to modify the loss function used in the clustering. In particular, most functions add an additional term for the attributes to Network Modularity and then cluster the network by standard network modularity maximization techniques using this new combined modularity [34], [32]. Thus, most intermediate integration techniques use the same modularity optimization techniques for a single network, but include an additional term for the attributes. There have also been some recently proposed techniques that adopt a matrix factorization approach to jointly factor both the network and attribute views to produce combined features which can be easily clustered by standard techniques (i.e. k-Means) [61], [54]. Recently, a late integration technique has been applied to attributed network clustering where a graph-based ensemble clustering technique is used on clusters from the attribute and network views [83]. Overall, while there have been late and intermediate integration techniques developed for attributed network clustering, most of the more successful attributed network clustering techniques still belong to the early integration paradigm [32]. While it is not definitively clear why certain multi-view clustering paradigms or techniques within the paradigms perform better than others, part of the problem is combining network and non-network views of data [32]. Since the network and non-network views of the data present very different features and can have different statistical properties and manifolds, combining information form these two views presents a distinct challenge.

Summary

The use of multi-view data can provide for distinct performance improvements in various machine learning tasks, including the unsupervised learning task of clustering. The use of multi-view data, however, presents new challenges including the fusion of the different views of the data for machine learning tasks like clustering [12]. The methods proposed for the fusion of multi-view data for clustering break into three major paradigms: early integration, intermediate integration, and late integration [12], [160]. Generally speaking, no one of these paradigms is *a priori* superior to any other paradigm, but rather the data environment often dictates what methods are plausible and, by corollary, which ones will be successful [160], [32]. That said, intermediate and late integration techniques are generally believed to be the superior paradigms across data environments due to their ability to exploit the two main principles of multi-view clustering, the complementarity and consensus principles [147], [12], [123]. The following table, Table 1.1, summarizes the paradigms of multi-view clustering.

Based on the current state of research on clustering multi-view data, there are some distinct gaps in the research. To date and the best of the author's knowledge only two works have proposed trying to cross the paradigms to produce a hybrid technique that utilizes late and intermediate integration information [70], [90]. Even the limited use of another paradigm's information in their technique gave a performance boost to their methods on their empirical tests [70], [90]. Thus, using a hybrid technique that can use concepts and information from both late and intermediate integration paradigms would seem to be a promising avenue for future research. Additionally, most multi-view clustering research has been for audio-visual-text, image, and medical/genetic data scenarios [160], [12], [99], [107]. One of the main differences

between these data scenarios and social-based data is that social-based data often comes with explicit network views of data, while these other data scenarios do not [32]. While there has been some research and development of techniques for social-based multi-view data, it has been limited to multi-layer/multiplex network clustering and attributed network clustering [32], [3]. It is not clear how the different multi-view clustering paradigms, and their associated techniques, might perform on more general types of social based data that could feature multiple network and non-network views.

Paradigm	Early Integration	Intermediate Integration	Late Integration
Description	Concatenate features from different views together and cluster as one view	Map all features to the same type/space and have an objective function defined over all views simultaneously	Cluster each view independently and then reconcile the clusters from the different views to produce the final clustering
Examples	- Feature vector concatenation - Network summation	- Tensor factorization - Expanded modularity maximization	- Cluster ensembling - Optimizing a cluster consensus measure
Strengths	- Simple to implement and conceptualize	- Generally considered to give the best results on a clustering task - Best exploit the complementary principle	- Can leverage specialized techniques for certain types of data - Often produce competitive results on the clustering task - Best exploit the consensus principle
Weaknesses	- Generally gives the worst results on a clustering task - Some views cannot be directly integrated	- Often computationally expensive - Require some mapping or transformation of the data to incorporate all of it into one model - Models used are often data scenario specific	- Do not take into account complementary information between views

Table 1.1: Summary of the different paradigms used in multi-view clustering

Chapter 2

Hybrid Paradigm Clustering of Multi-view Social-based Data

In this chapter I will primarily investigate the use of hybridization of intermediate and late integration paradigms in multi-view clustering. As was highlighted in the first chapter in the literature review, there are two main paradigms for multi-view clustering: intermediate and late integration. The main benefit of intermediate integration techniques is that they can best take advantage of the complementary information between the views of the data. Their main disadvantage is that they often require transformation of the data to some kind of compatible format or space which may result in a loss of information that could aid in the task of clustering for some views. On the other hand, the main advantage of late integration techniques is their ability to leverage data-specific clustering algorithms for the different views of the data to produce clusters. However, late integration techniques can suffer in performance from the fact that they do not take into account complementary information between the views when clustering. Thus, a natural question arises: is it possible to use aspects of both intermediate and late integration paradigms to develop techniques that leverage the strengths of both of the paradigms and possibly mitigate their weaknesses?

To date, there has been very little work on trying to produce techniques that combine aspects of late integration and intermediate integration paradigms [70], [90]. However, the works that have done some limited hybridization have shown strong empirical performance in finding clusters from multi-view data. In particular the inclusion of some view specific cluster information tends to preserve useful information relating to the clusters in the data that can be lost by the process of transforming the different views to all of the same type or space [70]. Conversely, using intermediate level information in late integration cluster ensembling can help to overcome the view-specific idiosyncrasies that can manifest in view clusterings, and thereby improve late integration performance [90], [121]. So, in this chapter I will outline a general framework to produce hybrid techniques for multi-view clustering, and then describe several hybrid techniques. In particular, I will show a class of techniques that use view-produced clusters to augment intermediate integration techniques and show a class of techniques that use intermediate integration data representations to improve late integration techniques.

After detailing the hybrid techniques, along with some new late and intermediate integration techniques, I then go on to test all of these techniques along with several state-of-the-art inter-

mediate and late integration techniques. The techniques are evaluated on several, benchmark social-based multi-view data sets that represents many of the social-based data scenarios that occur. Following this empirical comparison I then go on to evaluate the various aspects of the data sets as they relate to multi-view clustering performance. The main contributions of this chapter are as follows:

- This chapter is the first work to present a new paradigm for multi-view clustering that combines late and intermediate integration paradigms. It proposes a systematic means of hybridization and presents several possible hybrid paradigm techniques derived from this general concept of hybridization.
- This chapter has the first major empirical test of multi-view clustering on many different types of social-based data; previous works in multi-view clustering have only considered one type of multi-view, social-based data (i.e. multi-layer networks or attributed graphs).
- This chapter uses the results of the empirical tests to analyze the properties of multi-view data sets that correlate to the success of multi-view clustering and their clusterability. The analyses provide the first comprehensive, practical, empirically-based guidance on which multi-view clustering techniques to use as well as a better understanding of when multi-view clustering will be successful.

2.1 Hybrid Paradigm Clustering Techniques

In this section I will outline the proposed methodology for hybridizing intermediate and late integration techniques. I will detail two main classes of techniques that derive from this model. The first class are intermediate integration techniques that use additional late integration information. So, this class would contain techniques which have a loss function defined for all of the view's graphs simultaneously but incorporates elements of the view clusterings. The second class are those techniques that use intermediate integration information to augment late integration techniques. This class of techniques follow a late integration approach but incorporate view information, like view graphs, into the ensembling procedure.

Since the focus of this work is on social-based, multi-view data and intermediate integration techniques are often data scenario specific (due to the need to define a particular clustering quality measure over all of the views of the data), the focus will be on those intermediate integration techniques than can use graph representations of data [147]. It should also be noted that non-network data can — and often are — converted to graph representations in clustering for better cluster identification [23], [147]. So, the process of clustering multi-view social-based data from the intermediate integration paradigm consists of converting each of the non-network modes to a graph and then using this collection of graphs combined with a suitable clustering quality function and optimization procedure to produce clusters. On the other hand, for late integration, each of the views of the data are clustered by whatever technique is meaningful for that view's type of data, and then the collection of view clusterings is used to derive a final clustering for the data. Given the different paradigm approaches to clustering multi-view data, a general hybridization of the paradigms would be to create *both* graphs and clusters from every view, and then use *both* of those representations of the data to produce the final clustering. The following figure, Figure

2.1, summarizes the overall process of hybridizing intermediate and late paradigm approaches to clustering multi-view, social-based data.

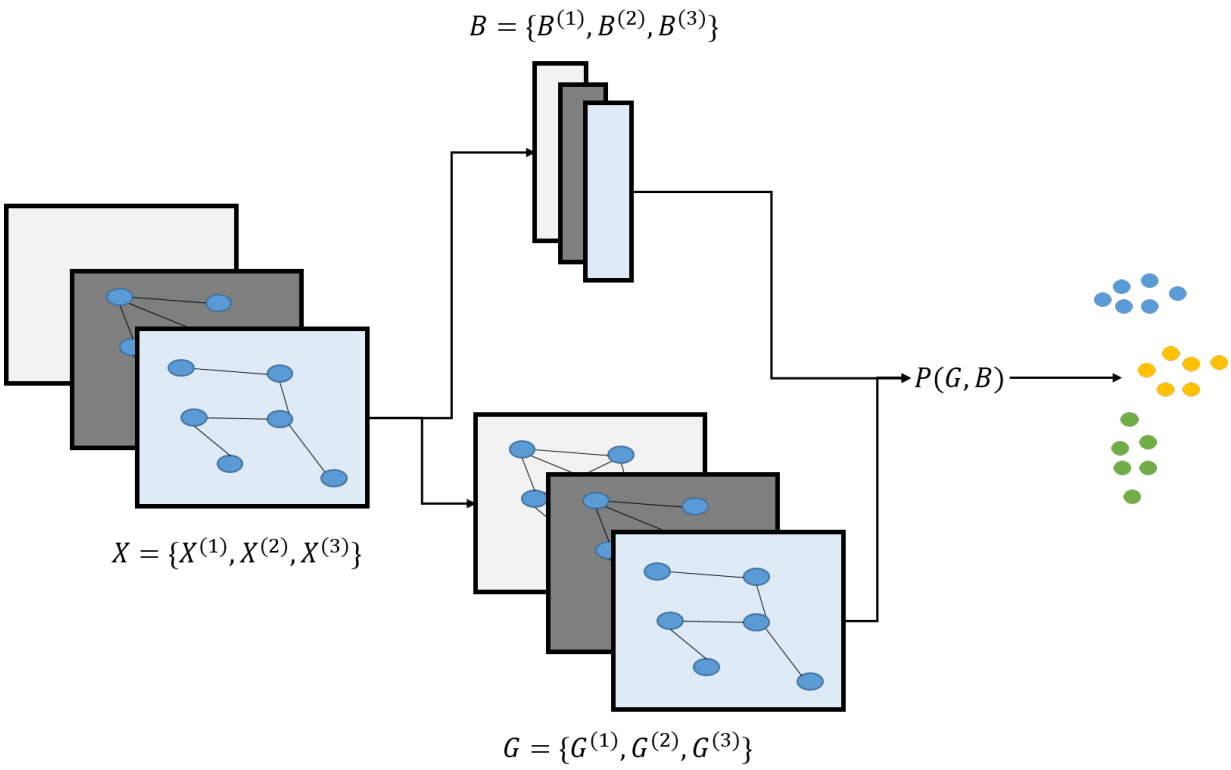


Figure 2.1: General methodology of hybridizing intermediate and late integration paradigm techniques. When presented with multi-view data (which could contain networks), X, one clusters each view by an appropriate technique and produces graphs from the non-network modes of the data. These two transformations of the data are combined in hybrid paradigm clustering technique, $P(G, B)$ to produce the final clusters of the multi-view data.

For each view of the data, two transformations are done. First, each view can be clustered by a suitable clustering technique. These clusters can then be transformed into a *cluster association matrix* B^v for each view of the data, v, of size the number of objects being clustered, n, by the number of clusters for that view, k^v, which has the property that all rows in B^v sum to one, $\sum_{j=1}^{k^v} B_{ij}^v = 1, \forall i = 1...n$. That is to say that each object being clustered has a probability of association with a certain view's clusters. Generally speaking, B^v will often be a *binary* cluster association matrix with only one nonzero element in each row equal to one due to the fact that most cluster ensembling is done with hard or crisp clustering in which each object can only belong to one cluster [18], [152]. Second, a graph can be derived from each view of the data, G^v. For the views that are already networks or graphs, no transformation of the data is required to produce G^v. For non-graph modes several techniques from graph learning or network inference can be employed to transform the data to a graph [23], [106]. After the transformation of the data into the formats previously described, this transformed data is then subsequently used to obtain the final clusters through either intermediate, late, or hybrid integration techniques.

2.1.1 Late Integration Information in Intermediate Integration Models

In this first class of techniques, late integration information — namely, the view clusterings — are used to augment intermediate integration techniques. In particular, these techniques employ the same general procedure developed in augmented network clustering in that they combine graph and attribute information directly to produce an augmented graph structure which can then be clustered as a graph [32]. This set up presents two main design choices: the first is the means by which a particular view's graph is combined with cluster labels. The second is how the final augmented graphs from each view are then clustered.

The two main means of combining graph information with the view clusterings are an additive model and a multiplicative model. The general form of the additive model is:

$$\alpha A^v + (1 - \alpha^v) C^{v\prime} \tag{2.1}$$

where A^v is the adjacency matrix of the graph from view v, $C^{v\prime}$ is a suitably sparsified co-association matrix. The co-association matrix is constructed from all of the view cluster association matrices, B, of all of the modes except the current mode:

$$C^v = \frac{1}{m-1} \sum_{i=1, i \neq v}^{m} B^i B^{i^T} \tag{2.2}$$

The co-association matrix is sparsified in this model as it is often much denser than the graph of the view and can thus overwhelm any information coming from A^v as well as be more difficult to cluster if it is not sparsified [81], [122]. α^v is a user-set parameter(s) which controls the amount of influence given to either the graphs from the different views or the clusterings coming form the views. The general form of the multiplicative model is:

$$A^v \odot C^v \tag{2.3}$$

where A^v is once again the adjacency matrix for a particular view and the C^v is the co-association matrix of of all the cluster association matrices, B, from all of the views except the current view, v. In this case, the co-association matrix is not sparsified, as sparsification will come inherently from the Hadamard product with the view's adjacency matrix. Thus, the only edges that will exist in the augmented graph will be those whose vertices share clusters within the views *and* are connected in the graph of the view.

There are several intermediate integration techniques which could be used to cluster the augmented graphs. In this work we focus on three in particular due to their strong empirical performance. The first is Spectral Integration, which was introduced by Tang in Liu in [123] for the purpose of finding clusters in multi-layer social networks and is a generalized version of Spectral Clustering [96]. In this method, the top k eigenvectors are extracted from each of the augmented view graphs' Laplacians and concatenated together to form a new feature matrix of size n by $k \times m$. This feature matrix then has Singular Value Decomposition applied to it as a view blending and denoising process to produce a final feature matrix of size n by k. This final feature matrix is then clustered by k-Means [123]. The second method is a co-learning inspired procedure referred to as cross-learning [16], [134]. For this method each augmented graph from each of the views is clustered to produce a new set of view clusters for each view, B^v_{t+1}. These

new view clusterings are then combined with the view graphs to produce new augmented graphs for each view. This process is repeated for a series of iterations and the final output of $B^v_{t_{final}}$ is then clustered by a clustering ensemble technique to produce the final clusters. The third method is to treat the task of clustering the augmented view adjacency matrices as a multiplex network clustering problem. In this case, multi-layer modularity maximization is used to cluster the view augmented graphs to produce the final clusters [93].

Direct Integration with Spectral Clustering

The first method within this class of methods is the *Direct Integration with Spectral Clustering* method. At a high level, after finding view-specific cluster association matrices, B^v, and creating graphs for each view, G^v, the view clusterings are integrated into the modal graphs to produce augmented graphs. These augmented graphs are then clustered by taking the top k eigenvectors corresponding to the k smallest eigenvalues of the graph Laplacians of each view's augmented graph. These eigenvectors are then concatenated together to form a new n by $k \times m$ feature matrix. Singular Value Decomposition is then performed on this feature matrix and the first k columns of the left singular matrix form the final feature matrix which is then clustered by k-Means. The pseudocode of the Direct Integration with Spectral Clustering procedure is detailed below, in Algorithm 1

Algorithm 1 Direct Integration with Spectral Clustering (DISC_A, DISC_M)

input:
- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- Integration model type: {additive, multiplicative}
- Graph-Cluster trade off parameters: $\alpha^v \in [0, 1]$ (only for additive model)
- Number of clusters, k

output: Cluster assignments
for $v = 1 : m$ **do**
  $C^v \leftarrow \frac{1}{m-1} \sum_{i=1, i \neq v}^{m} B^i B^{i^T}$
  if $additive$ **then**
    $C^{v\prime} \leftarrow prune(C^v, density(A^v)$
    $Aug^v \leftarrow A^v + (1 - \alpha^v) C^{v\prime}$
  else
    $Aug^v \leftarrow A^v \odot C^v$
  end if
  $L^v \leftarrow I - D^v_{inv} Aug^v$
  $U^v \leftarrow eig(L^v)[:, 1 : k]$
end for
$U \leftarrow concatenate(U^1, ..., U^m)$
$V \leftarrow SVD(U)[:, 1 : k]$
$clusters \leftarrow kMeans(V, k)$
return $clusters$

For each view of the data, the algorithm computes the augmented adjacency matrix, Aug^v based on whether the view clusterings are incorporated by a multiplicative model or an additive model. In the case of the additive model, the additional step of pruning the the cluster-association graph is performed. Each augmented adjacency matrix is used to compute a random walk Laplacian, $I - D_{inv}^v Aug^v$, where D_{inv}^v is a diagonal matrix where each diagonal entry is the inverse of the sum of its corresponding row (i.e. vertex degree) in the adjacency matrix, $\sum_{j=1}^{n} A_{ij}, \forall i = 1...n$. The random walk Laplacian is used instead of a standard or normalized Laplacian due to recent research demonstrating that the random walk Laplacian is more suitable for clustering [133]. Each views random walk Laplacian can then be decomposed by any standard procedure to produce eigenvectors, $eig()$, of which the top k are kept. Each of the view's eigenvectors, U^v, are then concatenated together and and SVD is run on the resulting matrix, U. The top k left singular vectors, V, from the Singular Value Decomposition operation, $SVD()$, are then clustered by kMeans, $kMeans(V, k)$, to produce the final clusters.

An important operation done in the additive model is the $prune()$ procedure which sparsifies the cluster-association matrices, C^v. Typically, the pruning procedure is a global procedure done by picking a value in the interval $[0, 1]$ and then removing all edges from the co-association graph that have a weight beneath that value [23], [80]. Some more recent works use a combination of local edge additions and global thresholding to get better results in cluster ensembling [121], [122]. In view of the success of these works, the pruning procedure used in this work is a global-local thresholding. At a high-level, it works by the user selecting the maximum sparsity that the graph should be pruned back to, rather than the edge weights that should be pruned. It then determines what the global threshold, τ, would be for the edge weights to be removed. For each vertex that has an edge with a weight less than τ, only the lowest weighted edges are removed such that the vertex keeps a fraction of its edges equivalent to the maximum sparsity. The pruning algorithm is described in detail by the following psuedocode, Algorithm 2:

Algorithm 2 Global-Local Pruning of Graphs

input:
- Adjacency matrix to be pruned, C
- Maximum sparsity of the pruned graph, d

output: Pruned Adjacency Matrix

$\tau \leftarrow quantile(nonzero(C), d)$
for $i = 1 : n$ **do**
  $minimum \leftarrow min((nonzero(C_i))$
  if $minimum < \tau$ **then**
    $\tau_{local} \leftarrow quantile((nonzero(C_i), d)$
    $C_i[C_i < \tau_{local}] \leftarrow 0$
  end if
end for
return C

Based on the maximum sparsity supplied, d, the algorithm calculates the edge weight, τ, where d percent of edges are beneath that edge weight. The algorithm the goes through each row in the adjacency matrix (vertex in the graph) and once again uses the quantile to find the

edge weight, τ_{local}, such that d percent of that vertex's edges fall below τ_{local}. The first advantage of this method of pruning is that it prevents isolates (i.e. a vertex is not connected to any other vertices in the graph) from forming in a co-association matrix which can strongly affect the performance of subsequent graph clustering techniques. Secondly, the algorithm still maintains a global threshold so that strongly connected vertices (i.e. those objects which often fall into the same clusters together, regardless of view) do not experience any local thresholding. Once the thresholding is complete the method finally returns the newly pruned graph.

An important variation of the Direct Integration Spectral techniques previously described is to use a *regularized spectral model* instead of the additive or multiplicative models. In this case, the graph Laplacian of each of the view graphs, L^v, is now regularized by the view clusterings. The inspiration for this model comes from constrained clustering research, where one wants to include externally-determined constraints about certain objects that should or should not be clustered together [84]. In this model, we follow the model proposed by Liu et al. in [84] and now do the eigendecompositon of $L^v + \lambda C^v$, where λ is the user-set regularization parameter that controls how much influence the view clusterings should have. So, the final Direct Integration Spectral method is detailed in the following algorithm, Algorithm 3.

Algorithm 3 Regularized Multi-View Spectral Clustering (DISC_R)

input:

- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- Regularization parameter: λ
- Number of clusters, k

output: Cluster assignments

for $v = 1 : m$ **do**

$C^v \leftarrow \frac{1}{m-1} \sum_{i=1, i \neq v}^{m} B^i B^{iT}$

$L^v \leftarrow I - D^v_{inv} A^v$

$U^v \leftarrow eig(L^v + \lambda C^v)[:, 1:k]$

end for

$U \leftarrow concatenate(U^1, ..., U^m)$

$V \leftarrow SVD(U)[:, 1:k]$

$clusters \leftarrow kMeans(V, k)$

return $clusters$

Taken all together the spectral methods present three distinct techniques for combining late integration information, in the form of view clusterings, into an intermediate integration technique. These techniques work on the fundamental, and empirically successful, clustering ideas from Spectral Clustering, and its various extensions to multi-view data but with the inclusion of view-specific cluster information.

Direct Integration with Cross-Learning

The second method within this class of methods is the *Direct Integration with Cross-Learning* method. Much like the previous method, augmented matrices are constructed from the graphs of

each view, G^v, as well as the cluster co-associations from each view, C^v, by either an additive or multiplicative model. The major difference with this method is that now, instead of extracting the spectral qualities of the augmented graphs for clustering, the augmented graphs are re-clustered to produce a new set of view clusterings, B^v_{t+1}, which can then be used to create a new set of cluster co-associations from each view, C^v_{t+1}. This process is repeated for a series of iterations or until stability is reached in the cluster assignments across all of the views. The following psue-docode displays the the algorithm, Algorithm 4, for direct integration of view cluster information and view graphs in a cross learning scenario.

Algorithm 4 Direct Integration with Cross-Learning (DICL_A, DICL_M)

input:
- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- number of iterations: T
- Graph-Cluster trade off parameter: $\alpha^v \in [0, 1]$ (only for additive model)
- Graph clustering algorithm: f
- Bipartite Graph clustering algorithm: g

output: Cluster assignments
for $v = 1 : m$ **do**
  $B^v_1 \leftarrow B^v$
end for
for $t = 1 : T$ **do**
  for $v = 1 : m$ **do**
    $C^v \leftarrow \frac{1}{m-1} \sum_{i=1, i \neq v}^{m} B^i_t B^{iT}_t$
    if $additive$ **then**
      $Aug^v \leftarrow A^v + (1 - \alpha^v) C^v$
    else
      $Aug^v \leftarrow A^v \odot C^v$
    end if
    $clusters \leftarrow f(Aug^v)$
    $B^v_t \leftarrow$ create_association_matrix $(clusters)$
  end for
end for
$B_{final} \leftarrow concatenate(B^1, ..., B^m)$
$clusters \leftarrow g(B_{final})$
return $clusters$

In each iteration of the algorithm, an augmented matrix is constructed for every view, Aug^v, using either an additive or multiplicative model. This augmented matrix is then clustered by any suitable graph clustering technique. In this work a modularity maximization technique such as Louvain or Leiden is employed for this procedure do to their speed and proven empirical performance [15], [129]. The result of this graph clustering then becomes that view's clusters for the next iteration of the cross-learning procedure. In this way, the algorithm uses the cluster structures from all of the views to influence the cluster structure arising from any particular view

and that view's graph. Once the algorithm has completed all of the cross-learning iterations, it then concatenates all of the view cluster association matrices, B^v, into a final object-by-clusters matrix, B_{final}. the final step is to use a cluster ensembling technique to get the final cluster assignments. Since B_{final} can also be a bipartite graph between objects and clusters, the algorithm uses the BGPA algorithm and clusters B_{final} as a bipartite graph [47]. To perform the bipartite clustering in this work, the bipartite modularity maximization procedure, BiLouvain, is utilized due to its scalability and proven empirical performance [155].

One variation to the model used in the cross-learning procedure is to have the view clusters become vertices in the augmented graphs. This model follows techniques developed in the attributed graph clustering to incorporate nodes' categorical attributes into networks [30], [32]. Since there are often fewer clusters than there are objects being clustered and each object will belong to at least one cluster per view, the view clusters function as hub vertices within the view graph, which will then bias the subsequent clustering of the augmented view graph toward the view clusters. The following algorithm, Algorithm 5, displays the pseudocode of this variation in the Cross-Learning techniques.

Algorithm 5 Graph Vertex Augmentation with Cross-Learning (DICL_AG

input:
- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- number of iterations: T
- Graph clustering algorithm: f
- Bipartite Graph clustering algorithm: g

output: Cluster assignments
for $v = 1 : m$ **do**
  $B_1^v \leftarrow B^v$
end for
for $t = 1 : T$ **do**
  for $v = 1 : m$ **do**
    $C^v \leftarrow concatenate(B^i \forall i = 1, ..., m, i \neq v)$
    $C^{v\prime} \leftarrow concatenate(C^{v^T}, zeros(k^v, k^v)^T$
    $Aug^v \leftarrow concatenate(Aug^v, C^v)$
    $Aug^v \leftarrow concatenate(A^{v^T}, C^{v\prime})^T$
    $clusters \leftarrow f(Aug^v)[1 : n, :]$,
    $B_t^v \leftarrow$ create_association_matrix $(clusters)$
  end for
end for
$B_{final} \leftarrow concatenate([B^1, ..., B^m)$
$clusters \leftarrow g(B_{final})$
return $clusters$

As with the direct integration algorithm, Algorithm 4, the graph vertex augmentation algorithm functions in much the same way. The main difference is that for each view, v, the the

augmented matrix, Aug^v, is created by appending a cluster association matrix, C^v, of all the views except the present views to the adjacency matrix of that view, A^v, to form a new adjacency matrix. The new augmented adjacency matrix will have the form of:

$$\begin{bmatrix} A^v & C^v \\ C^{vT} & 0 \end{bmatrix}$$

After constructing the augmented adjacency matrix, the algorithm then clusters the new matrix using a suitable graph clustering technique (i.e. Louvain). Only the clustering assignments for the objects are then carried forward to the next step of the algorithm. After the clustering step, the algorithm proceeds exactly the same as the other cross-learning algorithm including using a bipartite graph clustering of the concatenated view cluster association matrices to produce the final clusters.

In the cross learning techniques, the view clusterings are used in combination with the view graphs to iteratively improve the consensus between the view clusterings. The combination of view clusterings with the view graphs can be done in three possible models: an additive model, a multiplicative model, and a augmenting model whereby the clusters are added to the view graphs as nodes. After completing iterations of cross learning, the final clustering is done by a cluster ensembling technique applied to the clusterings coming from all of the views.

Direct Integration with Multiplex Clustering

The final method within this class of methods is the *Direct Integration with Multiplex Clustering* method. As with the previous methods in this section, this method uses the same additive and multiplicative models to integrate the view clusterings into the view graphs. Once integrated, the method then treats the problem of finding clusters from the augmented graphs as a multiplex network clustering problem [41]. One of the most successful methods to cluster a multiplex network is the modularity maximization technique originally proposed by Mucha et al. in [93]. So, within the proposed method of this section, we use the same modularity maximization technique to cluster all of the augmented graphs from all of the views. The following pseudocode, Algorithm 6, summarizes the algorithm.

In this algorithm, the final clustering is done through a multiplex modularity optimization technique. The particular form of the modularity is give as:

$$Q = \sum_{v=1}^{m} Q^v \tag{2.4}$$

That is to say the modularity value for a particular clustering of a multiplex network, Q, is a (possibly weighted) combination of the modularities that is produced by the clustering for all of the views of the data, Q_v. In the case of a view graph being undirected, the modularity of that view graph is given by:

$$Q^v_{undirected} = \sum_{i,j=1}^{n} (A^v_{ij} - \frac{degree^v_i \times degree^v_j}{2|E^v|})\delta(c_i, c_j) \tag{2.5}$$

Algorithm 6 Direct Integration with Multiplex Clustering (DIMC_A, DIMC_M)

input:
- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- Multiplex Modularity Maximization Algorithm: f
- Graph-Cluster trade off parameter: $\alpha^v \in [0, 1]$ (only for additive model)

output: Cluster assignments
for $v = 1 : m$ **do**
  $C^v \leftarrow \frac{1}{m-1} \sum_{i=1, i \neq v}^{m} B^i B^{iT}$
  if $additive$ **then**
    $C^{v\prime} \leftarrow prune(C^v, density(A^v)$
    $Aug^v \leftarrow A^v + (1 - \alpha^v) C^{v\prime}$
  else
    $Aug^v \leftarrow A^v \odot C^v$
  end if
end for
$clusters \leftarrow f(Aug^1, ..., Aug^m)$
return $clusters$

Where A^v is a particular view's adjacency matrix, $degree_i^v$ is the degree of vertex i ($\sum_{j=1}^{n} A_{ij}^v$), E_v is number of edges present in the view graph, and $\delta(c_i, c_j)$ indicates whether vertices i and j are in the same cluster. γ_v is a resolution parameter which controls the impact of the null model of edges which is used in determining a cluster's significance in terms of its internal and external edges [94]. Similar to the undirected case, in the case of a view graph being directed, the modularity of that view graph is given by:

$$Q_{directed} = \sum_{i,j=1}^{n} (A_{ij}^v - \frac{outdegree_i^v \times indegree_j^v}{2|E^v|}) \delta(c_i, c_j) \tag{2.6}$$

Where now $outdegree_i^s$ is the outdegree of vertex i in A^v, (i.e. $\sum_{j=1}^{n} A_{ij}^v$) and $indegree_j^s$ is the indegeree of vertex j in A^v, (i.e. $\sum_{i=1}^{n} A_{ij}^v$). Having defined the modularity measure being optimized for determining the community structure, any suitable modularity maximization technique can be used [101]. Overall, the general idea of this method is to integrate all of the late and intermediate integration into a collection of view networks, which are then a multiplex network, which can then be clustered by techniques for multiplex clustering.

2.1.2 Intermediate Integration Information in Late Integration Models

In the second class of techniques, the information used in intermediate network clustering — namely the view graphs — are used to improve the view cluster qualities for a late integration approach. In has been mentioned in previous chapters, one of the major shortcomings of late integration approaches is that they do not take into account complementary information between the views when doing the clustering; all view clusterings are done independently. That said,

late integration approaches can also take advantage of specialty clustering algorithms that are designed for specific types of data, like modularity maximization for networks or topic modeling for text. So, late integration techniques can avoid possible cluster information being lost in the conversion to a common data format (i.e. a graph) that is common in many intermediate integration techniques. Furthermore, the use of just the view clusters can also be a form of denoising of the original data [124]. So, in this section, we seek to use the advantages of late integration techniques, but also include information from the view graphs to add complementary information back into the view clusterings and thereby improve the performance of late integration techniques.

Cross-Network Diffusion Clustering

The first method in this class of the methods is called *Cross-Network Diffusion Clustering*. This technique builds on the ideas of similarity network fusion [134], [135]. This method uses a graph diffusion model combined with a cross-learning process in order to learn a similarity matrix between all of the objects which can then be clustered for the final clusters. The graph diffusion model is the Tensor Product Graph, given by:

$$S_{t+1} = \alpha P S_t P^T + (1 - \alpha) I \tag{2.7}$$

Where S is a similarity matrix between the objects, P is a sparse kernel graph (i.e. a k-Nearest Neighbor graph of the objects), I is an n-by-n identity matrix, and $alpha$ is a user defined parameter that controls how far the impact of diffusion reaches into the graph. Each of the matrices is row normalized. In this work the Tensor Product Graph diffusion model is used due to its demonstrated empirical performance and guaranteed convergence under certain assumptions like all of the matrices being row stochastic [146], [43].

As in the previous section and in Similarity Network Fusion, the different similarity matrices can be used with kernel graphs other than the view that the similarity matrix came from using the cross-learning method. Thus, the update model for a particular view becomes:

$$S^v_{t+1} = \alpha^v P^v \frac{\sum_{i=1, i \neq v}^{m} S^i_t}{m-1} P^{vT} + (1 - \alpha^v) I \tag{2.8}$$

In this model, to update the similarity matrix for a particular view, S^v, all of the similarity matrices from the other views are element-wise averaged together and then this matrix is used in the diffusion model with the kernel matrices from that view, P^v. Once a suitable number of diffusion steps have been run, the final similarity matrices are then averaged together and clustered as a graph or affinity matrix.

This method leaves some possible design choices regarding how to incorporate the view graphs and view clusterings. In this work I use the view graphs as the kernel graphs and have two different variations for the use of the view clusterings. In the first variation, the co-association matrices from the view clusterings can be used as the initial similarity matrices for each of the modes, S^v_0. Thus, similarity is the idea of being in the same clusters and this similarity is then fused across the modes using the view graphs. This fusion uses the view graphs to provide intermediate integration information back into the view clusterings so that the view clusterings

obtain some complementary information from across the views before the final clustering. In the second variation co-association matrices from the view clusterings can be used in place of the identity matrix I in the diffusion model. In the original diffusion model, the use of the identity matrix is to promote 'self-similarity' and the greatest impact of diffusion for a particular vertex to be from itself [146]. Thus, in this model using co-association matrices in the second part of the diffusion model ensures that learned similarities are continually biased by the view clusterings. All together, the following pseudocode, Algorithm 7, details the Cross-Network Diffusion algorithm with its possible variations.

Algorithm 7 Cross Network Diffusion Clustering (CNDC_C1, CNDC_C2, CNDC_I)

input:
- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- Integration model type: {first, second, none}
- Diffusion strength parameter: $\alpha^v \in [0, 1]$
- Graph-based or Similarity-based clustering algorithm: f

output: Cluster assignments
for $v = 1 : m$ **do**
  $C^v \leftarrow \frac{1}{m-1} \sum_{i=1, i \neq v}^{m} B^i B^{iT}$
  if $first$ **then**
    $S_0^v \leftarrow$ row_normalize(C^v)
  else
    $C_0^v \leftarrow$ row_normalize(C^v)
    $S_0^v \leftarrow I$
  end if
  $P^v \leftarrow$ row_normalize(A^v)
end for
for $t = 1 : T$ **do**
  for $v = 1 : m$ **do**
    if $first$ **then**
      $S_{t+1}^v = \alpha^v P^v \frac{\sum_{i=1, i \neq v}^{m} S_t^i}{m-1} P^{vT} + (1 - \alpha^v) I$
    else if $second$ **then**
      $S_{t+1}^v = \alpha^v P^v \frac{\sum_{i=1, i \neq v}^{m} S_t^i}{m-1} P^{vT} + (1 - \alpha^v) C^v$
    else
      $S_{t+1}^v = \alpha^v P^v \frac{\sum_{i=1, i \neq v}^{m} S_t^i}{m-1} P^{vT} + (1 - \alpha^v) I$
    end if
  end for
end for
$S_{final} \leftarrow \frac{\sum_{i=1}^{m} S_T^i}{m}$
$clusters \leftarrow f(S_{final})$
return $clusters$

This algorithm first converts the view adjacency matrices, A^v into diffusion kernel matrices,

P^v, by row normalizing them. Then, it creates view co-association matrices, C^v from each of the view clusterings and row-normalizes them as well. Then depending on the model type, (i.e. first or second) the initial similarity matrix for each view S_0^v is assigned as the identity matrix or the row-normalized cluster association matrix. From there, the diffusion is done for T iterations using the appropriate diffusion model (i.e. first or second). The final view diffusion matrices are then element-wise averaged together and the result is clustered by a graph-based or similarity-based clustering technique. Note, that it is also possible to have a diffusion model where none of the view clusterings are taken into account (none). This variation of the algorithm is solely an intermediate integration technique and uses just the graphs of all of the views to learn the similarity matrix between the objects being clustered.

The method proposed in this section uses view graphs to infuse intermediate level information into the view clusterings for better subsequent late integration clustering. This is done by using a diffusion model that iteratively adjusts the similarities between objects depending on their nearness in the view graphs. In this way, complementary information between the views is encoded into the learned similarity matrices which should then lead to better clusters. Within the method there are a few variations in how to include the view clusterings to include not including the view clusterings at all. The different models allow for the view clusterings to have different levels of impact on the subsequently learned similarities. Overall, this method leverages a cross-learning process with graph diffusion in order to combine useful information from the different views in order to identify clusters from the multi-view data.

Cross-View Influence Clustering

The other method in this class of methods also employs a diffusion process, but has a much different model for that diffusion process. *Cross-View Influence Clustering* uses an diffusion model inspired by Friedkin's Social Influence model [51]. The diffusion model is given by:

$$B_{t+1}^v = \alpha^v W^v B_t + (1 - \alpha^v) B_0 \tag{2.9}$$

Where B_t is the cluster association matrix from the previous iteration, W^v is the row normalized adjacency matrix of the graph for a particular view, and α^v is an influence trade-off parameter that controls the how much influence the neighbors of an object have on that object in a particular view. At each iteration, the cluster associations for each object are updated as a combination of the neighbors of that object in that view and by the the original cluster associations B_0. In this way, the model can infuse information from the different views through the view graphs into the view clusterings in order to provide more complementary information for the subsequent cluster ensembling task. To infuse this information across all of the views, after each iteration the resulting view cluster association matrices are averaged together to provide the cluster association for the next step:

$$B_{t+1} = \frac{\sum_{i=1}^{m} B_{t+1}^i}{m} \tag{2.10}$$

Following all of the diffusion iterations, the final cluster association matrices from each view can then be averaged together and the result clustered as a bipartite graph. The following pseudocode, Algorithm 8, details the Cross-View Influence Clustering algorithm.

Algorithm 8 Cross-View Influence Clustering (CVIC)

input:

- Adjacency for each view: A^v
- Cluster association matrices for each view: B^v
- Diffusion strength parameter(s): $\alpha^v \in [0, 1]$
- Number of iterations: T
- Bipartite Graph clustering algorithm: g

output: Cluster assignments

for $v = 1 : m$ **do**
  $W^v \leftarrow$ row_normalize(A^v)
end for
$B_0 \leftarrow concatenate(B^1, ..., B^m)$
$B_1 \leftarrow B_0$
for $t = 1 : T$ **do**
  for $v = 1 : m$ **do**
    $B^v_{t+1} = \alpha^v W^v B_t + (1 - \alpha^v) B_0$
  end for
  $B_{t+1} = \frac{\sum_{i=1}^{m} B^i_{t+1}}{m}$
end for
$B_{final} \leftarrow \frac{\sum_{i=1}^{m} B^i_T}{m}$
$clusters \leftarrow g(B_{final})$
return $clusters$

The algorithm begins by row-normalizing all of the view graphs and creating one cluster association matrix, B_0, by appending all of the cluster association matrices together. From their the algorithm performs an iteration of the diffusion for each view and then combines the results from each view together. This new cluster association matrix is then fed back into the diffusion model until the desired number of iterations is reached. Much like the previous diffusion based technique, Cross-Network Diffusion Clustering, this technique will also converge to stable cluster association matrices [51]. Once the iterations are complete the final cluster association matrix can be clustered as a bipartite graph. The clusters of this graph can be used to define both communities of objects but also communities of the original view clusterings.

The Cross-View Influence Clustering technique builds on the ideas of social influence. In this technique, the view clusterings are treated as the 'beliefs' and they are subsequently updated through interactions in the view graphs. So, complementary information between the views can be infused into the the original view clusterings by using local influence from the different view graphs. In this way the strength of association for an object to view cluster assignments is affected by the nearby objects from the different views, which allows for some complementary information between the views to affect the view clusterings.

The two techniques, along with their variations, presented in this section generally rely on diffusion processes to infuse additional information into view clusterings, and thereby improve the performance of late integration techniques. These view clusterings should then be more representative of the whole of the multi-view data set as they now contain some complementary information form across the views. The subsequent ensembling of these modified clusters should then perform better than just using the view clusterings by themselves due to their complementary information.

2.2 Empirical Testing

In this section I will use several benchmark, social-based data sets in order to empirically evaluate the hybrid paradigm techniques proposed in the previous section. The proposed techniques will be compared to several state of the art intermediate and late integration techniques at their ability to recover the given clusters from the benchmark data sets. The general set up of the empirical testing is as follows:

- Each view of each data set is clustered by an appropriate clustering technique.
- Each non-network view of the data set is transformed into a graph. The same graph is used across all methods that require graphs.
- The collection of each data set's clusters and graphs are then clustered by each multi-view technique twenty times. This is to account for the fact that many of the optimization routines for these techniques are stochastic.
- Each of the twenty results are then evaluated against the benchmark labels for each data set and the average performance is reported.

In the next section the data sets will be described in detail, including view cluster and view graph information. In the following section all of the techniques that are being used for comparison will be briefly described. Finally, the results of empirical testing will then be presented.

2.2.1 Data Set Information

For the empirical evaluation I used twelve different, social-based data sets. The data sets were chosen to encompass a wide array of scenarios that give rise to multi-view data. Since multi-view data can arise from many different types of social interactions, which can all be qualitatively different, it is important to see if there are any techniques that can work for social-based data generally or any that work for particular social scenarios. Since it is well established in clustering theory that different clustering functions will have different properties with inherent trade-offs, it is expected that certain certain types of clustering functions may work well for some data scenarios, but poorly for others [76], [7].

The data sets break into four types of scenarios that give rise to multi-view data: publication networks, social media, news stories, and multi-layer networks. Of the social media data sets, most of the data sets come from Twitter, however there is also representation of other social media sites as well. All but two of the data sets include both network and non-network views of the data. More traditional multi-view data sets of a text only data set and a multi-layer graph are also included in order to better evaluate the proposed methods across the full range of possible views arising from social-based data. The following table, Table 2.1, displays a summary of the different data sets.

Data set	Views	Number of Objects	Number of Clusters	Meaning of Clusters
Cora [113]	- Text (binary feature vector) - Network	2,708	7	General categories of the papers
CiteSeer [113]	- Text (binary feature vector) - Network	3,312	6	General categories of the papers
Flickr [66]	- Network - User tags	7,575	9	User Communities
BlogCatalog [66]	- Network - Blog keywords	5,196	6	User Communities
Wiki [144]	- Text (TF-IDF feature vector) - Network	2,405	19	General categories of the pages
3Sources [73]	- BBC (text term counts) - Reuters (text term counts) - The Guardian (text term counts)	169	6	News topic
AUCS [28]	- Network (lunch) - Network (Facebook) - Network (co-author) - Network (work) - Network (leisure)	61	9	Research group

Football [57] (Twitter)	- List - Text (term counts) - Network (follows) - Network (mentions) - Network (retweets)	248	20	Team membership
Olympics [57] (Twitter)	- List - Text (term counts) - Network (follows) - Network (mentions) - Network (retweets)	464	28	Sport affiliation
Politics IE [57] (Twitter)	- List - Text (term counts) - Network (follows) - Network (mentions) - Network (retweets)	348	10	Political party
Politics UK [57] (Twitter)	- List - Text (term counts) - Network (follows) - Network (mentions) - Network (retweets)	419	4	Political Party
Rugby [57] (Twitter)	- List - Text (term counts) - Network (follows) - Network (mentions) - Network (retweets)	854	8	National affiliation

Table 2.1: Benchmark, social-based data used in the empirical validation of the techniques. Most data sets consist of network and non-network modes of the data. An all text data set (3Sources) and an all network data set (AUCS) were also included in order to get a better empirical investigation across the types of social-based data that exists.

Since all of the multi-view methods evaluated in this section use either the clusters or the graphs, or both, from every view, the following tables summarize the view cluster qualities and the graphs form each of the views. The first table, Table 2.2, summarizes the clusters from each of the views across each of the data sets.

Data set	Mode	ARI	AMI	Without labels	Number of clusters	Best method to cluster	Actual number of clusters	Average ARI consensus between view clusters
Cora	network text	0.21 0.12	0.43 0.17	0.83 -0.02	107 7	Leiden LDA	7	0.05
CiteSeer	network text	0.1 0.11	0.25 0.14	0.89 0.002	466 6	Leiden LDA	6	0.05
Flickr	network attributes	0.09 0.56	0.11 0.56	0.25 -0.38	15 19	Leiden Leiden on kNN	9	0.06
BlogCatalog	network attributes	0.13 0.41	0.23 0.44	0.37 -0.04	8 6	Leiden LDA	6	0.04
Wiki	network	0.2	0.29	0.77	72	Leiden	17	0.22

	text	0.38	0.45	-0.085	17	LDA		
3Sources	BBC -text	0.16	0.34	0.02	6	Agglomerative	6	0.147
	Reuters -text	0.01	0.23	0.05	6	Agglomerative		
	Guardian -text	0.12	0.24	0.1	6	Agglomerative		
AUCS	lunch network	0.59	0.64	0.66	6	Leiden	9	0.247
	facebook network	0.27	0.21	0.34	33	Leiden		
	coauthor network	0.18	0.17	0.76	44	Leiden		
	work network	0.54	0.55	0.46	5	Leiden		
	leisure network	0.45	0.45	0.57	20	Leiden		
Football (Twitter)	list	0.4	0.56	-0.04	19	LDA	20	0.396
	text	0.02	0.08	-0.05	18	LDA		
	follows	0.49	0.74	0.45	11	Leiden		
	mentions	0.78	0.87	0.67	18	Leiden		
	retweets	0.59	0.72	0.69	29	Leiden		
Olympics (Twitter)	list	0.45	0.56	-0.06	28	LDA	28	0.497
	text	0.09	0.25	-0.13	27	LDA		
	follows	0.57	0.78	0.48	11	Leiden		
	mentions	0.86	0.91	0.8	22	Leiden		
	retweets	0.85	0.89	0.84	47	Leiden		
Politics Ire (Twitter)	list	0.16	0.16	-0.05	7	LDA	7	0.315
	text	0.15	0.23	-0.04	7	LDA		
	follows	0.9	0.85	0.37	4	Leiden		
	mentions	0.85	0.78	0.53	12	Leiden		
	retweets	0.75	0.7	0.66	49	Leiden		
Politics UK (Twitter)	list	0.81	0.79	0.05	4	LDA	5	0.402
	text	0.14	0.15	-0.03	5	LDA		
	follows	0.97	0.92	0.4	4	Leiden		
	mentions	0.67	0.65	0.29	13	Leiden		
	retweets	0.64	0.59	0.36	43	Leiden		
Rugby (Twitter)	list	0.33	0.51	-0.09	15	LDA	15	0.506
	text	0.17	0.3	0.08	14	LDA		
	follows	0.63	0.67	0.53	14	Leiden		
	mentions	0.36	0.61	0.67	19	Leiden		
	retweets	0.35	0.6	0.7	44	Leiden		

Table 2.2: Cluster properties and performances for each view of each of the data sets. Note, that in many cases the number and quality of clusters differs significantly between network and non-network modes of the data. Furthermore, while the network measure of clustering performance, Network Modularity, is generally indicative of the clustering performance on the network views, the Silhouette Index is not indicative of the performance of clustering the non-network views.

In order to evaluate how well the view clusters match the given, benchmark labels both the Adjusted Rand Index (ARI) and the Adjusted Mutual Information (AMI) were used to compare the cluster labels to the given labels [69], [132]. Generally speaking, there is a wide disparity of view cluster qualities relative to the benchmark labels, both within and between data sets. Some data sets like Cora and 3Sources have no particular view cluster well, while many of the Twitter data sets have at least one interaction network that does cluster well. The average pairwise ARI between the clusters between each of the views was also recorded. This average pairwise ARI represents a measure of consensus between the individual view clusterings. There is a wide disparity in this average ARI consensus between the data sets with the Twitter data sets having the highest values. These results indicate that there can be a wide disparity in the views in terms

of their clusterability for recovering the benchmark labels. Based on the consensus principle of multi-view clustering, it would be expected that those data sets with a greater disparity in their view clusterings should also have worse multi-view clusterings.

In order to evaluate the performance of the view clustering techniques in the absence of any ground-truth labels I have has to adopt two different measures. If the view is a graph, then I use network modularity [94] to evaluate clustering performance. If the view is other than a graph, I use the Silhouette Index to evaluate clustering performance [109]. Both the modularity and the silhouette score do not correlate well with a view's clustering performance across the data sets. Nor do they provide much insight into determining the clusterability of views within a data set. Using these measures, one cannot definitively tell the clusterability of the various views of the data sets. This would suggest that the common approach clustering data by trying various numbers of clusters and/or clustering algorithms and choosing that which delivers the best silhouette score may not be suitable for these data sets. Additionally, for many of the non-network views the silhouette score is negative, indicating overlapping clusters and poor clustering performance. For these reasons, whenever the clustering method requires the number of clusters be specified (which is most of the non-network clustering techniques) I choose to simplify the clustering procedure and just set the number of clusters to the benchmark number of clusters. As a result, there is often a disparity in number of clusters between the network and non-network modes. However, this disparity is also likely, at least in part, a function of the differences in clustering structure between views. For example, the AUCS multi-layer data set, which only has network views, has different numbers of clusters within each view as found by a network clustering procedure which automatically determines the number of clusters. Thus, there is not clustering consistency between the views of the same data set which is one source of potential difficulty in clustering these multi-view data sets.

In terms of clustering the individual views, I tried various types of commonly available clustering techniques and used those which produced the best results in terms of the benchmark labels. For the network views, the clustering technique used the Leiden modularity maximization technique [129]. For the text views either Ward Aggolmerative Clustering (using the actual number of clusters) of Latent Dirichlet Allocation and selecting the most probable cluster (again, using the actual number of clusters) produced the best clusters from the non-network modes.

The second table, Table 2.3, summarizes the graphs from each of the views across each of the data sets. For each of the the non-network views, a symmetric k Nearest Neighbor graph (kNN) was constructed from the features of the view. To construct the symmetric kNN, each object is connected to k of its nearest neighbors to produce a graph and then only the reciprocated edges are kept. So, $e_{uv} \in E$ iff $u \in kNN(v)$ and $v \in kNN(u)$. For weighted graphs, the lowest edge weight is used as the reciprocated edge weight. The symmetric kNN can be formed from the adjacency matrix of the original kNN by doing a element-wise minimum between the adjacency and its transpose, by $A_{sym} = min(A, A^T)$. A symmetric kNN is used as it has been found to better reveal cluster structure in kNNs [110], [88], [89], [27]. For the value of k I used $k = \lceil \sqrt{n} \rceil$, which is generally regarded are producing effective graphs for clustering [88], [105].

Data set	Mode	Graph type	Graph density	Number of graph components	Number of graph isolates	Degree assorti-ativity	Clustering coefficient
Cora	network	directed	0.0007	78	0	-0.035	0.13
	text	undirected	0.008	1	0	0.404	0.14
CiteSeer	network	directed	0.0004	438	0	0.129	0.07
	text	undirected	0.007	4	3	0.378	0.13
Flickr	network	undirected	0.009	1	0	-0.217	0.33
	attributes	undirected	0.006	12	11	0.205	0.23
BlogCatalog	network	undirected	0.013	1	0	-0.01	0.12
	attributes	undirected	0.006	6	5	0.439	0.32
Wiki	network	directed	0.0029	45	0	0.0047	0.323
	text	undirected	0.012	98	97	0.534	0.48
3Sources	BBC -text	undirected	0.08	1	0	0.407	0.4
	Reuters -text	undirected	0.07	1	0	0.504	0.48
	Guardian -text	undirected	0.07	3	2	0.479	0.39
AUCS	lunch network	undirected	0.11	2	1	0.004	0.66
	facebook network	undirected	0.068	30	29	0.0027	0.283
	coauthor network	undirected	0.01	44	36	0.017	0.109
	work network	undirected	0.106	2	1	-0.213	0.63
	leisure network	undirected	0.048	16	14	-0.01	0.302
Football (Twitter)	list	undirected	0.046	1	0	0.51	0.69
	text	undirected	0.017	66	58	0.62	0.3
	follows	directed	0.06	2	1	-0.019	0.37
	mentions	directed	0.05	2	1	0.046	0.38
	retweets	directed	0.02	15	14	0.062	0.28
Olympics (Twitter)	list	undirected	0.03	2	0	0.49	0.56
	text	undirected	0.02	56	47	0.6	0.33
	follows	directed	0.05	2	1	-0.017	0.4
	mentions	directed	0.04	4	3	0.078	0.42
	retweets	directed	0.02	26	24	0.049	0.36
Politics IE (Twitter)	list	undirected	0.03	7	6	0.52	0.49
	text	undirected	0.02	58	56	0.5	0.36
	follows	directed	0.14	1	0	-0.082	0.48
	mentions	directed	0.05	6	5	-0.147	0.39
	retweets	directed	0.03	42	41	-0.152	0.34
Politics UK (Twitter)	list	undirected	0.03	26	21	0.48	0.47
	text	undirected	0.01	93	84	0.59	0.21
	follows	directed	0.16	2	1	-0.079	0.53
	mentions	directed	0.08	8	7	0.0207	0.33
	retweets	directed	0.04	39	38	0.067	0.26
Rugby (Twitter)	list	undirected	0.025	15	14	0.45	0.56
	text	undirected	0.01	201	186	0.69	0.29
	follows	directed	0.05	7	6	0.019	0.41
	mentions	directed	0.046	7	6	0.16	0.41

	retweets	directed	0.017	28	27	0.089	0.31

Table 2.3: Graph properties for all of the views across all of the data sets. As with the view clusterings in the previous table, there are often significant differences in topology between the the network view graphs and the non-network view graphs.

There is a high degree of variability of view graphs both between and within data sets. Typically, the graphs constructed from the non-network views have different topologies than the network view graphs. Across most of the data sets, the non-network view graphs are often denser, have higher degree assortiativity, are symmetric, and have fewer components and isolates. However, this pattern does not hold across all of the data sets, as most of the Twitter data sets reverse this pattern. All of these topological differences are partly related to the fact that the non-network views have their graphs constructed by a kNN procedure. However, these differences are not solely related to the network and non-network view differences. For example. the text only data set (3Sources) has differences in topology between the view graphs and all of these graphs were constructed by the same method with the same parameters. Thus, while there are distinct differences between the network and non-network views graphs, these differences can only partly a function of the graph learning technique for the non-network views. So, the differences in graph topologies between views in a data set is function of the differences in views and the processes that form the different views. Thus, it is unlikely that the graphs from every view will line up topologically and this lack of alignment will contribute to the difficulty of producing one set of cluster labels for all of the graphs.

The graphs in these social-based data sets have topological features that can present difficulties for clustering that other data scenarios do not have. For example, many of the data sets will have one or more view graphs that feature isolates. These isolates are normal in social-based data scenarios; they often represent individuals that are not interacting in a particular view. For example not all users will engage in re-tweeting or re-posting behavior. This is in contrast to other data scenarios, like image processing, where there will never be isolates since all images interact in all of the views. Additionally, the graphs from social-based data can also be undirected, directed, weighted, or unweighted all within the same data set. In other data scenarios all of the view graphs are typically of only one type (i.e. unweighted and undirected). So, social-based, multi-view data scenarios present new challenges for multi-view clustering techniques as they have much more difficult graph topologies to cluster and have a large degree of heterogeneity between views of the data than do other multi-view data scenarios.

2.2.2 Compared Methods

In this section all of the methods that will be tested across the social-based data sets are described. Several state-of-the-art late and intermediate integration techniques that are used in the empirical tests are described in this section. The is section will also detail the user-set parameter settings for all of the proposed techniques form the methods section of this chapter.

The intermediate and late integration multi-view clustering techniques from other works I investigated are as follows:

Intermediate Integration Techniques

- **CG**: Consistent Graph [82] aims to learn both consistent and inconsistent graphs across graphs from from multiple views of the data. The method does this using graph representations of each of the views of the data and a unified optimization framework. The final consistent graph output is clustered via spectral clustering for the final clusters.
- **ETL-MSC**: Essential Tensor Learning for Multi-view Spectral Clustering [139] uses a regularized tensor decomposition method along with some other tensor operations to get a lower-dimensional representation of the data. Each mode of the data is converted into a graph and then each graph into a probability transition matrix, which are then stacked to produce a tensor. The output of the decomposition is then clustered by spectral clustering for the final clusters.
- **NFC-CCE**: Network Fusion for Composite Community Extraction [55] uses a Non-Negative Matrix Factorization for a graph of each view and across each of the views in a combined model to produce a low-dimensional cluster indicator matrix which can be clustered via k-Means. Each view of the data is converted into a graph for input into this method. I used $\alpha = 0.0001$ which was determined by hand tuning the parameter to best fit the social-based data.
- **SPSL**: Self-Paced Spectral Learning [150] iteratively clusters a weighted combination of the graphs from each of the views, and updates the weight for each view's graphs depending on how well that particular view graph does with the clusters obtained by partitioning the weighted combination of all of the view graphs.
- **ResK**: RESCAL Factorization with K-Means Clustering [130] creates a tensor from all of the graphs from each of the views and then uses RESCAL tensor factorization of this tensor to produce a cluster indicator matrix. The method then clusters the cluster indicator matrix by k-Means to produce the final clusters.
- **SFI**: Spectral Feature Integration [123] computes the random-walk normalized Laplacian for each of the view graphs. It then takes the top eigenvectors from each view's Laplacians, concatenates them, and finally uses SVD on the concatenated matrix to produce the final set of feature vectors. It then clusters those feature vectors via k-Means to get the final clusters. I each decomposition, only k vectors were taken forward from the decomposition.
- **MSIM_C**: In this technique [70] row and column similarity matrices are learned in a co-learning set up, across all of the views. The row-similarity matrices are also clustered in every iteration to produce an co-association matrix, which is then used to enhance the row-similarity matrix. After a set number of iterations are run, the final row-similarity matrix is then clustered as a graph to produce the final clusters. It should be noted that this technique does not require each of the views to be converted to a view graph, however, in empirical tests I have found universally better performance if it is given the view graphs instead of the raw data matrices from each view. Each test ran for four iterations, which was the suggested number of iterations from the paper.

Late Integration Techniques

- **CSPA+**: Cluster-based Similarity Partitioning Algorithm [119] computes the co-association matrix from all of the view clusterings. I have then incorporated a global-local prun-

ing procedure and iterative refinement of the co-association matrix, both of which have been shown to improve the performance of co-association matrix-based methods [80], [90], [122]. Additionally, I also use modularity maximization techniques to cluster the co-association matrices due to their superior empirical performance in terms of accuracy and speed [80].

- **BGPA+**: Bipartite Graph Partitioning Algorithm [18], [47] is a cluster ensembling technique that forms a bipartite graph from every object and every cluster across all modal clusterings. This bipartite graph can then be clustered to get final cluster assignments. I improve upon existing algorithms by using a bipartite graph-specific clustering algorithm rather than a generic graph clustering algorithm. In particular, I use a BiLouvain algorithm for clustering the object-by-clusters bipartite graph [1].
- **LWMC**: Locally Weighted Meta-Clustering [64] is a cluster ensembling technique that clusters a specially-weighted cluster-to-cluster graph in order to produce meta-clusters which can then be used to determine the final cluster assignments. The weighting in the cluster-to-cluster graph is done by determining the joint entropy for each cluster with respect to every other cluster, across all modes, and represents how well that particular cluster is at capturing an actual cluster in the data.
- **LWBG**: Locally Weighted Bipartite Graph clustering [64] is a cluster ensembling technique which clusters a specially weighted bipartite graph of objects-to-clusters from all of the modes. The weighting in the object-to-cluster bipartite graph is done by determining the joint entropy for each cluster with respect to every other cluster, across all modes, and represents how well that particular cluster is at capturing an actual cluster in the data.
- **GP-MGLA**: Graph Partitioning with Multi-Granularity Link Analysis [65] is a cluster ensembling technique that creates a weighted graph of all of the objects and the clusters. The weight in the graph varies based on the type of the vertices of the edge (i.e. whether its a object to cluster or cluster to cluster). This weighted graph of objects and clusters is then clustered as a standard weighted graph. I set $\alpha = 0.5$ and $\beta = 2$ as was done in the method's original paper.
- **MCLA+** Meta-Clustering Algorithm [119] is a cluster ensembling technique that computes a weighted cluster-to-cluster graph in order to produce meta-clusters which can then be used to determine the final cluster assignments. In this implementation, I use Louvain [2] instead of METIS to cluster the cluster-to-cluster graph and weight the cluster-to-cluster graph by the mutual information between clusters rather than the Jaccard Index, as these modifications produced better results.
- **DREC**: Dense Representation based Ensemble Clustering [156] uses a sparse graph learning technique to learn a similarity matrix from the cluster association matrices from each view. In order to improve performance, the algorithm also uses a slimming procedure, by which all of the objects that are clustered together in all of the views are grouped together and treated as a new object. The final similarity matrix is then clustered by spectral methods. I used a grid search over the values of $[0.001, 0.01, 1, 10, 100]$ for the lambda

[1] https://scikit-network.readthedocs.io/en/latest/tutorials/clustering/louvain.html#Bigraphs
[2] https://scikit-network.readthedocs.io/en/latest/tutorials/clustering/louvain.html

regularization value, and selected that one which lead to the best performance in ARI and AMI.

In addition to those intermediate and late integration methods from other works, I also tested some intermediate and late integration techniques that are natural extensions of the techniques already proposed in this work. These techniques are described below:

- **IPMMC & LPMMC**: Intermediate Paradigm and Late Paradigm Multiplex Modularity Clustering. These techniques are a natural extension of the direct integration methods proposed in this work. Instead of combining the view clusterings into the view graphs and clustering these as a multiplex network, just use either the view graphs (IPMMC) or the cluster association matrices folded into graphs (LPMMC) in the multiplex clustering. The intermediate integration version, (IPMMC) is somewhat related to a technique for attributed network clustering in that the technique maps the attributes to a graph [32].
- **CNDC_I**: Cross Network Diffusion Clustering with no late integration information. As was mentioned in the section describing the CNDC technique, it is possible to not use any view clusterings in the diffusion equation and just have each view begin with and identity matrix for the staring similarity matrix.

For the proposed techniques relying on diffusion, (i.e. CNDC and CMIC), each were run for 30 iterations and used an $alpha$ value of 0.9 for all views, which is generally agreed upon as an appropriate setting for diffusion models [43]. For the direct integration with an additive model, the $alpha$ value was set to 0.5 for all views in all tests. Finally, for the direct integration techniques using cross-learning, cross-learning was done for 10 iterations.

2.2.3 Performance Across All Techniques and Data Sets

In this section I present the results of all of the methods being applied to all of the data sets. Since many of the techniques in this work have stochastic elements, each technique was run on each data set twenty times and the average scores are reported. The tables of results are broken out the data set types: publication networks, social media, single type of data, and Twitter. For each data set both the ARI and AMI are reported and the three highest results are shaded in green and the best result is in bold. In the first table, Table 2.4 are the results for the publication network data sets.

Data Set	Cora ARI	Cora AMI	CiteSeer ARI	CiteSeer AMI
GP-MGLA	0.22	0.28	0.15	0.17
CSPA+	0.12	0.17	0.12	0.14
BGPA+	0.23	0.32	0.15	0.17
LWBG	0.19	0.3	0.1	0.16
MCLA+	0.18	0.24	0.15	0.17
LWMC	0.21	0.22	0.18	0.19
DREC*	0.12	0.17	0.11	0.14
LPMMC	0.12	0.17	0.12	0.14
DISC_A*	0.15	0.21	0.15	0.17
DISC_M*	-0.004	0.004	0.11	0.25

DISC_R*	0.23	0.28	0.14	0.16
DICL_A	0.25	0.32	0.21	0.23
DICL_M	0.19	0.25	0.1	0.14
DICL_AG	0.36	0.4	0.32	0.31
DIMC_A	0.24	0.29	0.12	0.13
DIMC_M	0.25	0.43	0.11	0.25
MSIM_C	0.22	0.33	0.04	0.16
CVIC	0.35	0.45	0.23	0.24
CNDC_C1	**0.44**	**0.49**	**0.43**	**0.41**
CNDC_C2	0.4	0.45	0.26	0.26
CNDC_I	0.42	**0.49**	0.33	0.35
SFI*	0.01	0.19	0.03	0.047
SPSL*	0.0003	0.01	0.008	0.02
CG*	0.36	0.42	0.14	0.18
ResK*	0.03	0.17	0.03	0.16
IPMMC	0.33	0.4	0.37	0.37
NF-CCE*	0.25	0.34	0.28	0.3
ETL-MSC*	0.32	0.46	0.42	**0.41**
Best Individual Mode	0.32	0.17	0.11	0.25

Table 2.4: Multi-view clustering performance on publication networks. The top three techniques for each data set are highlighted in green and the top performing techniques is in bold font. The techniques are list in three different sections. From top to bottom they are late integration, hybrid integration, and intermediate integration. Techniques wirh an asterisk (*) by them indicate that the number of clusters must be supplied to the algorithm. The hybrid integration techniques of CNDC_C1 and the intermediate integration technique of ETL-MSC are some of the best performing on this data scenario.

Generally, the CNDC methods provide the best performance in terms of recovering the benchmark labels for the publication data sets. In particular, the CNDC_C1 model wherein the starting similarities for the diffusion process are the co-association matrices created from the view clusterings. The intermediate integration techniques of ETL-MSC and just using multiplex Leiden clustering on the view graphs (IPMMC) were also competitive. Turning now to the social media data sets, the following table, Table 2.5, displays the results.

Data Set	Flickr ARI	Flickr AMI	BlogCatalog ARI	BlogCatalog AMI	Wiki ARI	Wiki AMI
GP-MGLA	0.4	0.43	0.31	0.36	0.23	0.38
CSPA+	0.2	0.23	0.4	0.423	0.23	0.35
BGPA+	0.51	0.52	0.44	0.46	0.25	0.4
LWBG	0.56	0.57	0.15	0.25	0.25	0.38
MCLA+	0.27	0.31	0.3	0.33	0.23	0.37
LWMC	0.31	0.35	0.27	0.3	0.23	0.36
DREC*	0.27	0.32	0.41	0.43	0.23	0.39
LPMMC	0.09	0.15	0.38	0.45	0.21	0.34
DISC_A*	0.53	0.54	0.2	0.29	0.29	0.45

DISC_M*	0.01	0.04	0.28	0.34	-0.0007	0.11
DISC_R*	0.53	0.54	0.24	0.29	0.25	0.42
DICL_A	0.33	0.38	0.45	0.47	0.23	0.37
DICL_M	0.36	0.41	0.37	0.37	0.21	0.39
DICL_AG	0.29	0.33	0.27	0.26	0.31	0.46
DIMC_A	0.14	0.19	0.4	0.47	0.2	0.33
DIMC_M	0.54	0.56	0.41	0.49	0.2	0.4
MSIM_C	0.05	0.09	0.36	0.42	0.17	0.3
CVIC	0.58	0.58	0.51	0.52	0.31	0.44
CNDC_C1	0.1	0.18	0.62	0.61	0.24	0.38
CNDC_C2	0.57	0.58	0.29	0.37	0.24	0.36
CNDC_I	0.2	0.29	0.4	0.39	0.14	0.29
SFI*	0.004	0.04	0.02	0.05	0.03	0.14
SPSL*	2.90E-07	3.30E-07	-2.60E-05	-1.84E-05	-0.001	0.003
CG*	0.49	0.54	0.49	0.54	0.29	0.44
ResK*	0.013	0.12	0.05	0.2	0.04	0.36
IPMMC	0.57	0.64	0.63	0.62	0.29	**0.49**
NF-CCE*	0.46	0.51	0.4	0.44	0.24	0.46
ETL-MSC*	**0.63**	**0.65**	**0.74**	**0.71**	**0.34**	**0.49**
Best Individual Mode	0.56	0.56	0.41	0.44	0.29	0.45

Table 2.5: Multi-view clustering performance on social-media data. The top three techniques for each data set are highlighted in green and the top performing techniques is in bold font. The techniques are list in three different sections. From top to bottom they are late integration, hybrid integration, and intermediate integration. Techniques with an asterisk (*) by them indicate that the number of clusters must be supplied to the algorithm. ETL-MSC is a clear front-runner for this data scenario, although the simple technique of IPMMC also has strong performance.

The intermediate integration technique of ETL-MSC performed the best across all of the social media data sets. Using multiplex Leiden on the view graphs (IPMMC) was again competitive across nearly all of the data sets tested. Some of the hybrid integration techniques, particularly those of the late integration with intermediate integration information (i.e. CNDC and CVIC) were also competitive. However, the performance of these techniques could vary quite substantially between the data sets. For example, CNDC_C1 produces competitive results on BlogCatalog, but produces some of the worst results across all methods on Flickr. In the third table, Table 2.6, the results for data sets that are only of one data type (i.e. only text or only networks) are displayed.

Data Set	3Sources ARI	3Sources AMI	AUCS ARI	AUCS AMI
GP-MGLA	0.31	0.41	0.49	0.55
CSPA+	0.04	0.22	0.59	0.62
BGPA+	0.42	0.47	0.59	0.64
LWBG	0.3	0.41	0.43	0.36
MCLA+	0.38	0.45	0.55	0.63
LWMC	0.36	0.44	0.53	0.6

DREC*	0.33	0.3	**0.67**	**0.8**
LPMMC	0.34	0.43	0.57	0.69
DISC_A*	0.24	0.4	0.66	0.75
DISC_M*	0.5	0.59	0.51	0.57
DISC_R*	0.2	0.4	0.49	0.56
DICL_A	0.4	0.47	0.45	0.5
DICL_M	0.46	0.66	0.62	0.68
DICL_AG	**0.76**	0.73	0.59	0.62
DIMC_A	0.37	0.45	0.58	0.61
DIMC_M	0.6	0.66	0.61	0.67
MSIM_C	0.59	0.58	0.14	0.11
CVIC	0.53	0.56	0.6	0.63
CNDC_C1	0.68	0.69	0.66	0.7
CNDC_C2	0.65	0.65	0.63	0.67
CNDC_I	0.73	0.73	0.66	0.7
SFI*	0.63	0.66	0.56	0.59
SPSL*	0.62	0.7	0.53	0.59
CG*	0.65	0.73	0.6	0.71
ResK*	0.26	0.48	0.59	0.7
IPMMC	**0.76**	**0.74**	0.57	0.69
NF-CCE*	0.34	0.52	0.63	0.74
ETL-MSC*	0.46	0.57	0.1	0.2
Best Individual Mode	0.16	0.34	0.59	0.64

Table 2.6: Multi-view clustering performance on the single data type data sets (text only and multiplex network). The top three techniques for each data set are highlighted in green and the top performing techniques is in bold font. The techniques are list in three different sections. From top to bottom they are late integration, hybrid integration, and intermediate integration. Techniques with an asterisk (*) by them indicate that the number of clusters must be supplied to the algorithm. The intermediate integration algorithms of IPMMC and CNDC_I show strong performance as does the late integration technique of DREC.

For the text only data set, the best method was IPMMC, while for the multiplex network it was the late integration technique of DREC. The multiplex network data set was one of only four data sets where a late integration technique produced competitive results. Some of the hybridized techniques also produced competitive results, such as DICL_AG and DISC_A. Finally, in the last table, Table 2.7, are the results for the Twitter data sets.

Data Set	Football (Twitter) ARI	Football (Twitter) AMI	Olympics (Twitter) ARI	Olympics (Twitter) AMI	Politics Ire (Twitter) ARI	Politics Ire (Twitter) AMI	Politics UK (Twitter) ARI	Politics UK (Twitter) AMI	Rugby (Twitter) ARI	Rugby (Twitter) AMI
GP-MGLA	0.49	0.73	0.56	0.78	0.76	0.74	0.86	0.79	0.52	0.66
CSPA+	0.5	0.75	0.57	0.78	0.88	0.82	0.96	0.9	0.68	0.69
BGPA+	0.53	0.75	0.55	0.77	0.78	0.7	0.77	0.74	0.55	0.66
LWBG	0.65	0.79	0.72	0.85	0.74	0.68	0.83	0.73	0.45	0.63
MCLA+	0.43	0.67	0.56	0.54	0.59	0.59	0.44	0.56	0.44	0.57
LWMC	0.39	0.64	0.54	0.74	0.79	0.7	0.86	0.76	0.45	0.58
DREC*	0.78	0.88	**0.9**	**0.94**	0.91	0.84	0.86	0.82	0.52	0.63
LPMMC	0.51	0.76	0.56	0.77	0.88	0.82	0.96	0.91	0.64	0.68

DISC_A*	**0.83**	**0.89**	0.85	0.91	0.81	0.8	0.78	0.76	0.53	0.67
DISC_M*	0.47	0.73	0.56	0.76	0.45	0.6	0.9	0.84	0.34	0.54
DISC_R*	0.79	0.87	0.76	0.85	0.56	0.68	0.58	0.67	0.34	0.58
DICL_A	0.35	0.66	0.46	0.74	0.87	0.81	0.96	0.9	0.71	0.7
DICL_M	0.73	0.86	0.74	0.87	**0.92**	**0.87**	0.87	0.87	0.44	0.65
DICL_AG	0.44	0.71	0.51	0.75	0.89	0.83	0.78	0.77	0.51	0.63
DIMC_A	0.69	0.84	0.65	0.83	0.87	**0.87**	0.98	0.94	0.69	**0.71**
DIMC_M	0.59	0.81	0.57	0.79	0.9	0.86	0.98	0.94	0.64	0.69
MSIM_C	0.07	0.21	0.24	0.51	0.64	0.6	0.77	0.82	0.36	0.45
CVIC	0.51	0.75	0.57	0.8	0.83	0.77	0.96	0.9	**0.72**	0.7
CNDC_C1	0.36	0.63	0.49	0.72	0.83	0.79	0.92	0.87	0.63	0.65
CNDC_C2	0.53	0.76	0.55	0.79	0.75	0.73	0.97	0.93	0.64	0.66
CNDC_I	0.72	0.78	0.63	0.77	0.64	0.67	0.92	0.87	0.65	0.67
SFI*	0.52	0.73	0.4	0.69	0.32	0.51	0.7	0.64	0.32	0.54
SPSL*	0.79	0.87	0.77	0.88	0.75	0.82	0.98	0.96	0.54	0.66
CG*	0.71	0.79	0.85	0.89	0.65	0.67	0.64	0.63	0.4	0.56
ResK*	0.004	0.09	0.02	0.25	0.017	0.09	0.01	0.15	-0.0005	0.18
IPMMC	0.65	0.83	0.54	0.77	0.89	0.86	**0.99**	**0.97**	0.65	0.68
NF-CCE*	0.25	0.45	0.66	0.76	0.73	0.7	0.56	0.53	0.26	0.51
ETL-MSC*	0.7	0.78	0.83	0.87	0.44	0.55	0.67	0.58	0.36	0.53
Best Individual Mode	0.78	0.87	0.86	0.91	0.9	0.85	0.97	0.92	0.63	0.67

Table 2.7: Multi-view clustering performance on Twitter social media data sets. The top three techniques for each data set are highlighted in green and the top performing techniques is in bold font. The techniques are list in three different sections. From top to bottom they are late integration, hybrid integration, and intermediate integration. Techniques with an asterisk (*) by them indicate that the number of clusters must be supplied to the algorithm. There is no clear one techniques that is best, but many of the intermediate integration techniques show strong performance. Also, these data sets typically feature a particular (usually the mentions or retweets) which clusters very well by itself.

For these data sets the competitive techniques vary across data set and across paradigms; there is no one technique or multi-view clustering paradigm that has superior performance for all of these data sets. The direct integration methods tend to have a lot of high performing techniques, but no one technique that is consistently good across all of the data sets. IPMMC and DREC also show strong performance but only on one or two data sets. Additionally, all methods generally show better performance on the Twitter data sets then on any of the other data sets. These data sets also typically have at least one view that has good clusters relative to the benchmark labels. In fact, for the Football and Olympics data sets, the best individual view's clusters are better than most of the multi-view clusters.

Overall Summary of Technique Performances

Across all of the test data sets, no one method is dominant. Also, across all of the data sets late integration techniques are often mediocre in performance whereas hybrid and intermediate integration techniques have better performance. Methods like ETL-MSC can have good performance on social media data sets, except twitter, and even publication networks, but then very poor performance for Twitter. Similarly, methods like the CNDC family of techniques can have strong performance in publication networks and all text data sets, but very poor performance for some of the social media data sets. Additionally, some techniques, like just using multiplex clustering on the view graphs (IPMMC) are often not the best but are often competitive across nearly every data set. To further understand the comprehensive performance of the different techniques, we turn to Table 2.8, which provides a summary across all techniques and data sets.

Method	Average ARI	STD ARI	Average AMI	STD AMI	Within Top Three of ARI	Within Top Three of AMI	Best of ARI	Best of AMI
GP-MGLA	0.442	0.217	0.523	0.213	0	0	0	0
CSPA+	0.441	0.306	0.508	0.281	0	0	0	0
BGPA+	0.481	0.197	0.550	0.191	0	0	0	0
LWBG	0.448	0.252	0.509	0.228	0	0	0	0
MCLA+	0.377	**0.150**	0.453	**0.165**	0	0	0	0
LWMC	0.427	0.220	0.490	0.205	0	0	0	0
DREC*	0.509	0.305	0.555	0.295	3	3	2	2
LPMMC	0.448	0.291	0.526	0.281	0	0	0	0
DISC_A*	0.502	0.283	0.570	0.263	3	3	1	1
DISC_M*	0.344	0.277	0.448	0.290	0	0	0	0
DISC_R*	0.426	0.223	0.525	0.223	1	1	0	0
DICL_A	0.473	0.247	0.546	0.212	1	1	0	1
DICL_M	0.501	0.273	0.585	0.263	1	1	1	1
DICL_AG	0.503	0.212	0.567	0.203	2	2	1	0
DIMC_A	0.494	0.290	0.555	0.284	2	3	0	0
DIMC_M	0.533	0.260	0.629	0.208	2	2	0	2
MSIM_C	0.304	0.246	0.382	0.223	0	0	0	0
CVIC	0.558	0.209	0.612	0.185	3	2	1	0
CNDC_C1	0.533	0.237	0.593	0.195	4	3	2	2
CNDC_C2	0.540	0.215	0.601	0.202	2	1	0	0
CNDC_I	0.537	0.236	0.583	0.209	3	2	0	1
SFI*	0.295	0.268	0.402	0.282	0	0	0	0
SPSL*	0.416	0.384	0.459	0.412	2	2	0	0
CG*	0.523	0.200	0.592	0.190	1	1	0	0
ResK*	0.089	0.173	0.246	0.183	0	0	0	0
IPMMC	**0.603**	0.212	**0.672**	0.182	**5**	**7**	2	**3**
NF-CCE*	0.422	0.181	0.522	0.146	0	2	0	0
ETL-MSC*	0.501	0.214	0.567	0.177	4	5	**3**	**3**

Table 2.8: Performance summary of the techniques across all of the data sets. The top three techniques for each summary measure are highlighted in green and the top performing techniques is in bold font. The techniques are listed in three different sections. From top to bottom they are late integration, hybrid integration, and intermediate integration. Techniques with an asterisk (*) by them indicate that the number of clusters must be supplied to the algorithm. The performance measures of ARI and AMI were averaged across all of the data sets and the how many times a technique was the best or in the top three for a data set is also recorded. Generally speaking, those techniques which had good average performance were not often among the best for a particular data set, while those techniques which were among the best for a certain class of data sets tended to have low average performance as they could be among the worst for other data sets. The one exception to this pattern is IPMMC.

Across all of the data sets, the IPMMC method, which is the application of multiplex Leiden clustering to the view graphs, has the best results on average. This same method is also frequently in the top three methods across all of the data sets and within the top three across all methods for the being the best method for a particular data set. The CNDC_C1 method, wherein the view clustering are used as the starting similarity graphs for the diffusion model, along with the

intermediate integration, tensor factorization technique, and the late integration method of DREC have good performance on some data sets but not others. As a result, they are the best or at least in the top 3 for some data sets, but have low performance in terms of ARI and AMI on average. CNDC_C1 shows strong performance on the publication network and text only data sets. ETL-MSC shows strong performance on the non-twitter, social media data sets that have only two views. And, DREC shows strong performance on the Twitter data sets, wherein at least one of the view clusterings also performs well. Finally, some methods like CVIC have good average performance across all of the data sets, but are rarely the best method and are only occasionally within the top three performers for any given data set. Thus, it would seem from these results that while some methods may be good on average, they can be outperformed by other methods on specific data scenarios. Additionally, those methods which do feature superior performance on some data sets often have very poor performance on others, which makes there performance on average mediocre.

To get a better sense of the performance of the different techniques across the different data sets, the following figure, Figure 2.2, displays a color plot of the relative performance on each technique, for each measure across all of the data sets. Green indicates better performance on that data set and measure and red indicates worse performance and on that data set and measure.

From the color map, one can again observe similar conclusions to what has been mentioned previously. Namely, for many techniques, there can be high amount of variance in performance between the different data sets. For example ETL-MSC shows very good performance (among the best three) for publication networks and some of the social network sites, but then very poor performance for the Twitter data sets. Whereas, techniques like SPSL performed very poorly on the publication networks and social media sites — likely due to the components present in the view graphs — but fair to good performance on the Twitter data sets. It is also interesting to note that the late integration techniques, with a few exceptions, seem to be generally worse than either the hybrid or intermediate integration techniques. So, no one technique is dominant across all of the social-based data scenarios.

With these summary results detailed per method, we now turn to the summary of performance across paradigms. The following table, Table 2.9, displays the average performance of all of the methods from across the different paradigms.

Multi-view Paradigm	Macro Average ARI across Paradigm	STD ARI across Paradigm	Macro Average AMI across Paradigm	STD AMI across Paradigm
Late Integration	0.446	0.0386	0.514	0.033
Hybrid Integration	**0.4758**	0.0793	**0.551**	0.0712
Intermediate Integration	0.423	0.1645	0.5053	0.133

Table 2.9: Summary performance statistics across all techniques for the three paradigms. The hybrid integration techniques have the best average performance, followed by intermediate and then late integration techniques. However, many intermediate integration techniques did have good performance, but some did have very bad performance which resulted in a lot more deviation in their performance then other paradigms.

When the results from all methods across the three different paradigms are averaged together, the hybrid integration paradigm methods perform the best, followed by intermediate integration and late integration. While this does seem to indicate that the hybrid paradigm is empirically

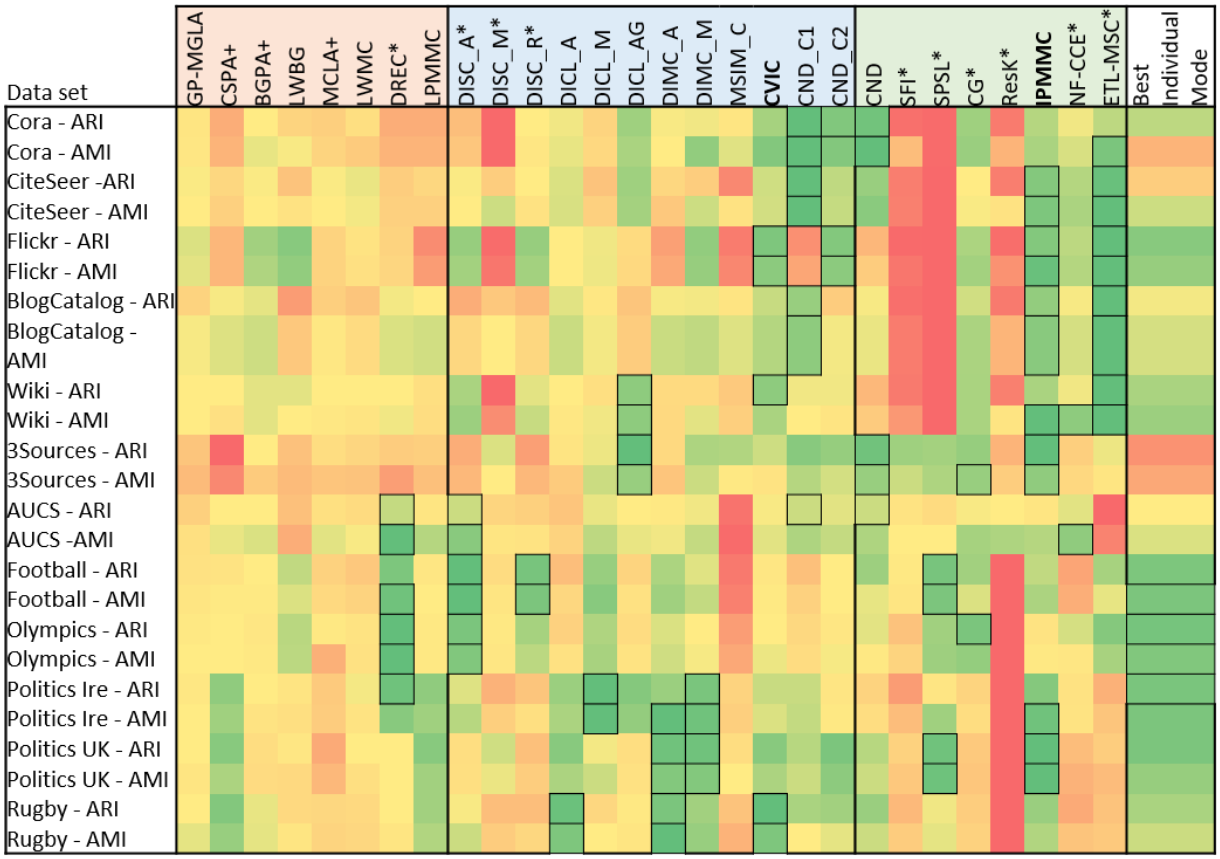


Figure 2.2: Qualitative performance overview of the different multi-view clustering techniques. Green indicates a better performance, relative to the other techniques and red indicates a worse technique. Color squares with a bolded border indicate that technique in the column achieved one of the top three performances on that data set. Across the top, the techniques are broken into late integration (light red shading), hybrid integration (light blue shading), and intermediate integration (light green shading). Techniques with an (*) indicate that that technique requires the number of clusters to be specified. Finally, techniques in bold font are those that had the highest average performance across all of the data sets.

superior to the other data sets for social-based data, it should be noted that the standard deviation in method performances is also much higher for intermediate integration techniques than for the other two paradigm's techniques. This high variation is a result of some methods, like ETL-MSC performing very well on some of the social media data sets, and other methods like SPSL or ResK performing very poorly on some of the data sets like social networks or publication networks. So, while the hybrid paradigm techniques tend to be better on average, depending on the data scenario, a properly chosen intermediate integration paradigm technique can give competitive, if not superior, performance.

2.2.4 Data Set Clusterability

One of the main results from the empirical testing is that clustering performance not only varies across all of the published and proposed techniques, but also across all of the data sets. Some

data sets like the publication networks have uniformly worse results than the Twitter data sets. Taken together, the results would suggest that the different data sets have different clusterability. In the theoretical work of traditional clustering there are many notions as to what makes a data set clusterable [2], [14], [7]. Following the theoretical guidance outlined in [2], the different clusterings produced by the different techniques can be used to analyze the clusterability of the data sets. Namely, I assume that each of the techniques can produce a 'suitable' clustering of the data set and I then compare these clusterings with their ARI and AMI values to the ground truth labels. In this way, both how clusterable the data set is and how well the data set's clusterability relates to the benchmark labels can be observed. The following table, Table 2.10, displays the average and standard deviations of the ARIs and AMIs for all of the data sets, across all of the methods.

Data set	Measure	Average across techniques	Std. across techniques
Cora	ARI	0.221	0.125
	AMI	0.295	0.132
CiteSeer	ARI	0.175	0.115
	AMI	0.208	**0.097**
Flickr	ARI	0.322	0.212
	AMI	0.360	0.201
BlogCatalog	ARI	0.350	0.173
	AMI	0.388	0.156
Wiki	ARI	0.210	**0.091**
	AMI	0.359	0.111
3Sources	ARI	0.461	0.185
	AMI	0.539	0.142
AUCS	ARI	0.543	0.134
	AMI	0.603	0.153
Football	ARI	0.535	0.207
(Twitter)	AMI	0.715	0.185
Olympics	ARI	0.593	0.186
(Twitter)	AMI	0.763	0.138
Politics Ire	ARI	0.717	0.208
(Twitter)	AMI	0.713	0.159
Politics UK	ARI	**0.802**	0.214
(Twitter)	AMI	**0.777**	0.176
Rugby	ARI	0.499	0.166
(Twitter)	AMI	0.609	0.108

Table 2.10: Average and standard deviations of clustering performance across all techniques for each of the data sets. The top three highest averages and lowest standard deviations for both ARI and AMI are highlighted in green and the top of each is in bold. The Twitter sets see the best performances on average, while the social media data sets see the least variability in performances across techniques.

Generally the twitter data sets see the highest performance relative to the benchmark labels while the the social media data sets see the lowest variability in performance. Interestingly,

those data sets with low variability in performance also tend to have low average performance, like Wiki or CiteSeer. This would suggest that these data sets are more easily clusterable in the sense that they have less variability in the clusters found by the different methods, but that the information used to form the clusters (i.e. the two different views) are not actually that informative for finding the benchmark clusters. In fact, for Cora and CiteSeer, this conclusion has been reached by other works that have bench-marked from these data sets [150], [143]. Conversely, the Twitter data sets have much higher average ARI and AMI values, but also higher variability in the clustering performances of the various techniques. However, from looking back at the other results tables, much of this variability is a direct result of one or two methods like MSIM_C, NF-CCE, or ResK doing very poorly on these data sets. Thus, these results would seem to indicate two things. First, that the Twitter data sets are actually easily clusterable and that the views used to form the clusters are actually informative to the cluster structure. Second, that not all techniques are suitable for any kind of social-based data. So, overall, there are differences in the clusterability of the different data sets with the Twitter data being some of the most clusterable of the benchmark data.

2.2.5 View Qualities and their Impact on Multi-view Clustering

Having observed the differences in clusterability of the data sets, another question arises as to what properties of the data sets could give rise to these performance differences across all techniques. In particular, I will explore in this section the differences in performance of the techniques as they relate to the view graphs and the view clusterings.

Effect of View Graphs

One of the critical components of the intermediate integration and hybrid integration techniques are the view graphs. In order to evaluate the view graphs, several metrics that are known to effect the clustering performance of view graphs are analyzed. The first metric is the average across all views of the percent of difference between the number of components of a graph and the number of clusters in the data set. It is generally believed that these numbers should be approximately the same, and so the more deviation there is, the more likely the graph structure performs poorly in recovering the clusters [89]. So, a higher Average percent difference between the number of components and the number of clusters, the lower the performance of multi-view clustering techniques should be. Another metric which could affect performance is the average across all views of the number of isolates relative to the number of vertices in the graphs. The affect of having a higher number of isolates is the same as that as having more components than clusters in a graph but even more so, since an isolate has no ties to any other vertex. A third metric would be whether the view graphs are heterogeneous with respect to the directedness of the graph; are the view are graphs all directed, undirected, or some combination of both. Since the different directedness of the graphs often require different clustering methods, it would be expected that those data sets with heterogeneous view graphs would have worse clustering performance. The following table, Table 2.11, summarizes these graph metrics for across the different data sets.

Data set	Average % Difference Between Clusters and Components	STD % Difference Between Clusters and Components	Average Isolate Ratio	STD Isolate Ratio	Heterogeneous graphs
Cora	0.884	**0.038**	**0.000**	**0.000**	Yes
CiteSeer	0.660	0.462	**0.000**	0.001	Yes
Flickr	0.569	0.452	0.001	0.001	No
BlogCatalog	**0.417**	0.589	**0.000**	0.001	No
Wiki	0.724	0.144	0.020	0.029	Yes
3Sources	0.722	0.192	0.004	0.007	No
AUCS	0.698	0.150	0.266	0.262	No
Football	0.739	0.290	0.060	0.100	Yes
Olympics	0.657	0.373	0.032	0.044	Yes
Politics Ire	0.543	0.433	0.062	0.072	Yes
Politics UK	0.720	0.232	0.072	0.079	Yes
Rugby	0.491	0.329	0.056	0.091	Yes

Table 2.11: Derived network statistics for the different data sets. The lowest values for the first four columns are highlighted in green and the lowest is bolded. Generally, a graph should have better cluster performance if it is lower in these metrics. The final column summarizes whether the data has all undirected graphs for its views or it has a combination of directed and undirected graphs across its views. Across the data sets, the Twitter data sets have more isolates, which are a result of a user not engaging in a certain behavior, like re-tweeting, but also have a number of graph components close to the number of clusters in the data set.

Generally, the Twitter data sets have lower average percent differences between the numbers of their view graph components and the number of clusters. The Twitter data sets are also make of the bulk of the heterogeneous data sets as well, as most of the observed networks are directed, while the graphs constructed from the text and list views are undirected. The publication networks have the lowest average isolates per number of vertices, which is a direct result of the data sets' construction, in that only publication with at least one other tie to the other nodes in the network are included.

In addition to the derived, graph related metrics, some mainstream, existing metrics are also known to be related to the presence and detectability of clusters within a graph. The first is the average across views of the average clustering coefficient of the graph [94]. Generally, the higher the clustering coefficient for a graph, the stronger the cluster or community structure within the graph is and the better a clustering algorithm should be able to work on the graph. The second is the average across views of the degree assortiativity. The degree assortiativity is the Pearson Correlation Coefficient of the degrees of connected vertices, can also believed to have an impact on the community structure present in networks [94], [13]. The following table, Table 2.14, presents the network metrics for all of the data sets.

Data set	Average of Clustering Coefficient	STD of Clustering Coefficient	Average Degree Assortiativity	STD Degree Assortiativity
Cora	0.135	**0.007**	0.185	0.310
CiteSeer	0.100	0.042	0.254	0.176

Flickr	0.280	0.071	-0.006	0.298
BlogCatalog	0.220	0.141	0.215	0.317
Wiki	0.402	0.111	0.269	0.374
3Sources	0.423	0.049	0.463	0.050
AUCS	0.397	0.239	-0.040	0.097
Football	**0.404**	0.166	0.244	0.297
Olympics	0.414	0.089	0.240	0.283
Politics Ire	0.412	0.069	0.128	0.350
Politics UK	0.360	0.136	0.216	0.299
Rugby	0.396	0.107	0.282	0.281

Table 2.12: Common network statistics used to ascertain whether there is a strong community structure within a network. The three best values are highlighted in green and the best value is bolded. The degree assortiativity statistics are not bolded as both assortiative networks and disassortiative networks could have distinct community structure. Generally, the twitter data sets possesses best metrics indicative of communities being present in their graphs.

As with the other graph metrics, the Twitter data sets generally contain the highest average clustering coefficient across their views. The degree assortiativity is generally positive — indicating vertices of the same degree typically connect to other vertices of the same degree — with the exception of the Flickr and the AUCS data sets. In both of these data sets, there is at least one highly disasortiative network. While it is not always clear whether a more assortiative or disassortiative network will cluster better, these data sets with a high variance in their degree assortiativiy, such as Cora, Wiki, BlogCatalog, and Flickr would be expected to have worse clustering performance since the connectivity patterns between the views will be more different.

With these graph metrics, it is now possible to compare the performance of the various techniques against these graph metrics. To do so, the correlation coefficient between the average ARI (only the ARI results are displayed since the correlation is comparable to the AMI, as the two measures of performance strongly correlate) and the different graph metrics was calculated. The following table, Table 2.13, summarizes the correlation coefficients between the multi-view clustering techniques' performance and the earlier derived graph metrics.

Measure	Average % Difference Between Clusters and Components	STD % Difference Between Clusters and Components	Average Isolate Ratio	STD Isolate Ratio	Heterogenous graphs
GP-MGLA	-0.183	0.080	0.390	0.441	0.219
CSPA+	-0.301	0.140	0.478	0.538	0.321
BGPA+	-0.322	0.165	0.450	0.492	-0.034
LWBG	-0.041	0.050	0.280	0.349	0.257
MCLA+	-0.221	0.044	0.598	0.649	0.008
LWMC	-0.118	0.011	0.443	0.466	0.198
DREC*	-0.168	0.123	0.465	0.534	0.216
LPMMC	-0.219	0.049	0.457	0.516	0.262
DISC_A*	-0.078	0.069	0.469	0.550	0.246
DISC_M*	-0.019	-0.027	0.436	0.471	0.050
DISC_R*	0.016	0.088	0.329	0.410	0.202
DICL_A	-0.345	0.145	0.268	0.307	0.194
DICL_M	-0.107	0.069	0.431	0.487	0.131

DICL_AG	0.053	-0.181	0.353	0.363	0.087
DIMC_A	-0.143	0.007	0.431	0.512	0.310
DIMC_M	-0.214	0.061	0.373	0.417	-0.019
MSIM_C	-0.133	-0.037	-0.020	-0.025	0.058
CVIC	-0.313	0.111	0.341	0.378	0.012
CND_C1	-0.148	-0.018	0.352	0.343	0.057
CND_C2	0.004	-0.137	0.384	0.412	0.017
CND_I	0.106	-0.179	0.414	0.475	0.123
SFI*	0.208	-0.302	0.525	0.572	-0.022
SPSL*	0.072	-0.121	0.397	0.479	0.246
CG*	-0.010	0.013	0.292	0.333	-0.129
ResK*	0.173	-0.370	0.784	0.701	-0.597
IPMMC	-0.305	0.217	0.199	0.233	-0.102
NF-CCE*	-0.238	0.228	0.466	0.406	-0.146
ETL-MSC*	-0.275	0.555	-0.498	-0.448	0.063

Table 2.13: Correlation between multi-view clustering techniques' performance and the derived graph metrics. It is generally expected that the correlation should be negative (i.e. as the graph metric increases in value or the graphs are heterogeneous, that performance should decrease). For the average difference between the number of components and clusters this expected negative correlation exists, but only weakly so, and the it doe snot exits for the ratio isolates present in the graphs. This suggests that the performance of the multi-view clustering techniques is not strongly effected by these network statistics.

The correlations between the multi-view clustering techniques' performance and the graph metrics present some surprising results. First, most techniques have a negative correlation between their performance and the percent difference between the average connected components and number of sub groups. This result should be expected, but it is surprising that this negative correlation holds for the late integration techniques — which do not use any of the view graphs — and not all of the intermediate integration techniques — which only use the view graphs. More surprising is the fact that the average isolate ratio is actually positive correlated with most techniques' performance. The expectation would be that these two measures should be negatively correlated. It is possible that because the the Twitter data sets, which had the highest average isolate ratios, also had the best individual view clustering performances, that having at least one view which clusters well is more important than the negative effect from the presence of isolates. A notable exception to this pattern is ETL-MSC which had much worse performance than the other techniques on the Twitter data sets. This would suggest that sensitivity to isolates is also technique dependent. Finally, having heterogeneous graphs is only consistently negatively correlated with the intermediate integration techniques. This result suggests that having heterogeneous graphs can indeed effect the performance of those techniques which rely on graphs. Turning now to the other graph metrics which can influence cluster structure and detection in graph, the following table, Table 2.12, displays the correlations of these metrics with the multi-view clustering techniques performance.

Measure	Average of Clustering Coefficient	STD of Clustering Coefficient	Average Degree Assortiativity	STD Degree Assortiativity
GP-MGLA	0.578	0.348	-0.220	0.213
CSPA+	0.493	0.482	-0.259	0.311
BGPA+	0.636	0.439	-0.296	0.118
LWBG	0.628	0.267	-0.233	0.252
MCLA+	0.773	0.493	-0.176	-0.096
LWMC	0.568	0.315	-0.183	0.092
DREC*	0.671	0.495	-0.152	0.142
LPMMC	0.600	0.430	-0.033	0.114
DISC_A*	0.668	0.466	-0.321	0.193
DISC_M*	0.562	0.456	0.131	-0.239
DISC_R*	0.544	0.413	-0.334	0.253
DICL_A	0.445	0.212	-0.054	0.184
DICL_M	0.677	0.417	-0.148	0.086
DICL_AG	0.586	0.077	0.132	-0.257
DIMC_A	0.601	0.426	-0.016	0.143
DIMC_M	0.635	0.323	-0.137	0.019
MSIM_C	0.358	-0.081	0.337	0.008
CVIC	0.543	0.306	-0.183	0.121
CND_C1	0.268	0.199	0.167	-0.216
CND_C2	0.543	0.206	-0.116	-0.132
CND_I	0.495	0.323	0.208	-0.308
SFI*	0.679	0.452	0.187	-0.453
SPSL*	0.710	0.338	0.156	-0.139
CG*	0.677	0.338	-0.031	-0.078
ResK*	0.239	0.536	-0.300	-0.750
IPMMC	0.487	0.234	0.049	-0.065
NF-CCE*	0.371	0.269	-0.487	-0.026
ETL-MSC*	0.045	-0.078	0.216	0.362

Table 2.14: Correlation between multi-view clustering performance and commonly used network statistics to characterize a network for having strong community structure. Generally, there is a strong positive correlation between the average clustering coefficient and the performance of multi-view clustering techniques. Thus, the stronger the community structure present within the view graphs the better the performance. For the other metric, there is no distinct correlation between the degree assortiativity and the multi-view clustering performance.

Having a higher average clustering correlation across all of the view graphs is always correlated with having better multi-view clustering performance. This is an expected result, however, it is surprising to see that the late integration techniques have just as strong of a correlation with the average clustering coefficient than do the intermediate integration techniques. This again is likely a result of the data sets that are high in clustering performance also having one or more views that cluster well. More surprising is that degree assortiativity has a negative correlation with late integration performance and a technique-dependent correlation for hybrid and intermediate integration techniques. The negative correlation between the late integration techniques

performance and disassortiativity likely comes from the fact that most of the data sets that had a low disassortiativity also tended to have somewhat better view clustering performance for at least one view (i.e. Flickr and Twitter data sets).

Overall, many of the graph quality metrics do not particularly correlate well with multi-view clustering performance. While some metrics like having the number of components in the view graphs be close to the number of clusters and the average clustering coefficient across all of the views do correlate strongly with clustering performance, other metrics like the average isolate ratio, surprisingly do not. That said some of these metric do correlate to performance for specific techniques, such as a greater presence of isolates leading to noticeably worse performance by techniques like ETL-MSC.

Effect of View Clusterings

In this section the effect of the view clusterings on the performance of the multi-view clustering techniques is investigated. In addition to the view graphs, the other main input to multi-view clustering techniques are the view clusterings. Also, earlier in the results section it was noted that data sets that have a particular view which cluster well relative to the benchmark labels (i.e. the Twitter data sets), also tend to have better multi-view clustering results, especially for late integration techniques. So, in this section, I will investigate the importance of the view clusterings in the performance of multi-view clustering techniques.

Based on the previous empirical testing results as well as the consensus principle in multi-view clustering, I analyzed three different aspects with regard to the multi-view clusters. The first is the performance of the best view clustering. The intuition behind this metric is that a data set which has a particular view cluster well should also have good multi-view clustering performance. The second is the average of the performances of the view clusterings. A data set which has a good average clustering performance across its views should also have good multi-view clustering performance. Finally, coming directly from the multi-view clustering principles is the consensus of clustering labels between the different views of the data sets. One can use the same clustering performance measures of ARI and AMI to compare how similar two views' clusterings are. A data set which has high consensus among its view clusterings should also have good multi-view clustering performance. The following table, Table 2.15, summarizes the multi-view clustering result for every method with the aforementioned view clustering results.

Method	Correlation between multi-view clusters and best view's clusters	Correlation between multi-view clusters and average of view's clusters	Correlation between multi-view clusters and average consensus between view's clusters
GP-MGLA	0.9375	**0.9421**	**0.7913**
CSPA+	0.9143	0.9240	0.7905
BGPA+	0.9068	0.8920	0.6992
LWBG	0.9176	0.9123	0.7466
MCLA+	0.7921	0.7972	0.7400
LWMC	0.8858	0.8923	0.7764
DREC*	0.9297	0.9170	0.7712

LPMMC	0.8483	0.8609	0.7841
DISC_A*	**0.9432**	0.9223	0.7681
DISC_M*	0.7162	0.7356	0.6797
DISC_R*	0.8911	0.8829	0.6885
DICL_A	0.8096	0.8161	0.7057
DICL_M	0.8920	0.8808	0.7224
DICL_AG	0.6034	0.6149	0.6552
DIMC_A	0.8740	0.8759	0.7832
DIMC_M	0.8356	0.8102	0.6433
MSIM_C	0.3781	0.4303	0.4295
CVIC	0.8551	0.8357	0.6626
CNDC_C1	0.4713	0.4999	0.4981
CNDC_C2	0.7700	0.7381	0.6221
CNDC_I	0.6256	0.6123	0.6155
SFI*	0.5882	0.6161	0.6719
SPSL*	0.7704	0.7572	0.7356
CG*	0.7101	0.6974	0.5228
ResK*	-0.1309	-0.0662	0.0508
IPMMC	0.7397	0.6981	0.5198
NF-CCE*	0.6537	0.6771	0.4767
ETL-MSC*	0.4163	0.4401	0.2050

Table 2.15: Correlation between cluster qualities across and between the views of each data set and the performance of the multi-view clustering techniques. The top three correlations are highlighted in green and the strongest correlation between the methods and the cluster quality metric is bolded. There is a strong positive correlation between a data set having a good view clusterings and multi-view clustering performance, especially for late integration techniques.

Generally, most methods have a high correlation with all three measures of view clustering quality. This result would suggest that most techniques rely on individual views having good cluster quality in order to have good clustering quality themselves. Some techniques like GP-MGLA which is a composite between CSPA and MCLA have a very high correlation between the view clustering qualities and the multi-view clustering quality. While most late integration techniques like GP-MGLA have high correlation in their clustering qualities to the clustering qualities of individual views, some hybrid integration techniques like DISC_A also have high correlation to the view clusterings. Somewhat surprisingly, most of the intermediate integration techniques (the exceptions being ETL-MSC and ResK) also high positive correlations to the view clustering performances. This result seems to suggest that the better any given view clusters are, the better the result of multi-view clustering will be, regardless of the paradigm. To further investigate the value of the quality of view clusterings and the the performance of multi-view clustering techniques, the following table, Table 2.16, gives the summary of the different paradigms of multi-view clustering and the different types of evaluating view clustering quality.

Measure	Correlation between multi-view clusters and best view's clusters	Correlation between multi-view clusters and average of view's clusters	Correlation between multi-view clusters and average consensus between view's clusters
Late Integration	0.892	0.892	0.762
Hybrid Integration	0.753	0.754	0.655
Intermediate Integration	0.547	0.554	0.475
All	0.734	0.736	0.634

Table 2.16: Correlation between cluster qualities across and between the views of each data set and the performance of multi-view clustering paradigms. Late integration techniques have the strongest correlation to the view clustering qualities followed by hybrid and than intermediate integration paradigm techniques. Generally, having a view that clusters well tends to correlate most strongly with multi-view clustering performance across the paradigms.

On average, late integration techniques have higher correlation with view clustering quality than do intermediate or hybrid integration techniques. That is to say when the view clusterings perform well, so too do the late integration techniques. This result should in some degree follow from the basic mechanism of late integration techniques; late integration techniques use the view clusterings the most out of all the paradigms and should therefore by more reliant on the view clusterings for their performance. It is interesting to note that there is little difference between having one view that clusters well and all views clustering well on average, and the performance of multi-view clustering techniques. This result would suggest that while it is beneficial to have all views provide information that is useful for clustering of the multi-view data, as long as one view performs well, multi-view clustering of the data will also perform well. Curiously, having consensus between all of the view's clustering results does not appear to be as strongly correlated as having one particular view that performs well in regards to multi-view clustering performance. This result would suggest that while meeting the consensus principle of multi-view clustering is important, it only matters in the context of having at least on view clustering well and only matters significantly for those techniques which employ view clusterings.

2.2.6 Comparison Between Hybridized and Non-Hybridized Techniques

One of the main contributions of this chapter is the introduction of the hybrid paradigm techniques in multi-view clustering. Some of these techniques are natural extensions of existing late or intermediate integration paradigm techniques. Thus, a natural to ask is how the hybridized versions of these techniques compares to the non-hybridized versions. In the first set of comparisons I compare the direct integration techniques (i.e. DICL and DIMC) to their non-hybridized, intermediate integration counterparts (i.e. IPMMC and SFI). Table 2.17 summarizes the difference in the performance metrics of ARI and AMI across all of the data sets for the direct integration techniques and their corresponding non-hybridized versions.

Data set	Measure	IPMMC & DIMC_A	IPMMC & DIMC_M	DISC_A & SFI	DISC_M & SFI	DISC_R & SFI
Cora	ARI	-0.09	-0.08	0.14	-0.014	0.22

	AMI	-0.11	0.03	0.02	-0.186	0.09
CiteSeer	ARI	-0.25	-0.26	0.12	0.08	0.11
	AMI	-0.24	-0.12	0.123	0.203	0.113
Flickr	ARI	-0.43	-0.03	0.526	0.006	0.526
	AMI	-0.45	-0.08	0.5	0	0.5
BlogCatalog	ARI	-0.23	-0.22	0.18	0.26	0.22
	AMI	-0.15	-0.13	0.24	0.29	0.24
Wiki	ARI	-0.09	-0.09	0.26	-0.0307	0.22
	AMI	-0.16	-0.09	0.31	-0.03	0.28
3Sources	ARI	-0.39	-0.16	-0.39	-0.13	-0.43
	AMI	-0.29	-0.08	-0.26	-0.07	-0.26
AUCS	ARI	0.01	0.04	0.1	-0.05	-0.07
	AMI	-0.08	-0.02	0.16	-0.02	-0.03
Football	ARI	0.04	-0.06	0.31	-0.05	0.27
(Twitter)	AMI	0.01	-0.02	0.16	0	0.14
Olympics	ARI	0.11	0.03	0.45	0.16	0.36
(Twitter)	AMI	0.06	0.02	0.22	0.07	0.16
Politics Ire	ARI	-0.02	0.01	0.49	0.13	0.24
(Twitter)	AMI	0.01	0	0.29	0.09	0.17
Politics UK	ARI	-0.01	-0.01	0.08	0.2	-0.12
(Twitter)	AMI	-0.03	-0.03	0.12	0.2	0.03
Rugby	ARI	0.04	-0.01	0.21	0.02	0.02
(Twitter)	AMI	0.03	0.01	0.13	0	0.04
Mean	ARI	-0.109	-0.070	0.206	0.048	0.131
Across All	AMI	-0.117	-0.043	0.168	0.046	0.123

Table 2.17: Performance comparison between the direct integration hybridized and non-hybridized techniques. The hybrid techniques that have directly analogous techniques for either intermediate or late integration are compared. A negative value indicates the non-hybridized techniques performed better on the data set and performance measure than the hybridized technique.

Across nearly all of the data sets, the hybridized technique of DIMC, wherein cluster information is directly incorporated into the view graphs and then these view graphs are clustered as a multiplex network, have worse performance than the non-hybridized counterpart, IPMMC, which is the multiplex clustering of just the view graphs. This result would suggest that the multiplex clustering of the view graphs does not benefit from the incorporation of view clusterings, even when a particular view or groups of views for a data set clusters well. In contrast, the comparison of the hybridized spectral direct integration techniques (DISC) with its non-hybridized counterpart (SFI) generally shows that hybridization had a positive effect on performance. For most of the data sets, and especially those with at least one view that clustered well, the spectral direct integration techniques often produce much better results than just using the same spectral technique on the view graphs alone. So, it would seem that hybridization in terms of incorporating view clusterings into an intermediate integration paradigm technique is only beneficial for certain techniques and under conditions of at least one view clustering performing well.

The other class of hybridized techniques that can be compared to non-hybridized techniques are the class of techniques that use the view graphs to diffuse view information into a late integra-

tion, cluster ensembling technique (i.e. CNDC and CVIC). These hybridized techniques can be compared to the late integration techniques that determine their final cluster assignments. In this way, it can be seen if incorporation of intermediate level information into the view clusterings (i.e. hybridization) is beneficial or not. The following table, Table 2.18, summarizes the comparisons between the diffusion-based, hybridized methods and their non-hybridized counterparts.

Data set	Measure	CNDC_I & CSPA+	CNDC_C1 & CSPA+	CNDC_C1 & CNDC_I	CNDC_C2 & CSPA+	CNDC_C2 & CNDC_I	CVIC & BGPA+
Cora	ARI	0.3	0.32	0.02	0.28	-0.02	0.12
	AMI	0.32	0.32	0	0.28	-0.04	0.13
CiteSeer	ARI	0.21	0.31	0.1	0.14	-0.07	0.08
	AMI	0.21	0.27	0.06	0.12	-0.09	0.07
Flickr	ARI	0	-0.1	-0.1	0.37	0.37	0.07
	AMI	0.06	-0.05	-0.11	0.35	0.29	0.06
BlogCatalog	ARI	0	0.22	0.22	-0.11	-0.11	0.07
	AMI	-0.033	0.187	0.22	-0.053	-0.02	0.06
Wiki	ARI	-0.09	0.01	0.1	0.01	0.1	0.06
	AMI	-0.06	0.03	0.09	0.01	0.07	0.04
3Sources	ARI	0.69	0.64	-0.05	0.61	-0.08	0.11
	AMI	0.51	0.47	-0.04	0.43	-0.08	0.09
AUCS	ARI	0.07	0.07	0	0.04	-0.03	0.01
	AMI	0.08	0.08	0	0.05	-0.03	-0.01
Football	ARI	0.22	-0.14	-0.36	0.03	-0.19	-0.02
(Twitter)	AMI	0.03	-0.12	-0.15	0.01	-0.02	0
Olympics	ARI	0.06	-0.08	-0.14	-0.02	-0.08	0.02
(Twitter)	AMI	-0.01	-0.06	-0.05	0.01	0.02	0.03
Politics Ire	ARI	-0.24	-0.05	0.19	-0.13	0.11	0.05
(Twitter)	AMI	-0.15	-0.03	0.12	-0.09	0.06	0.07
Politics UK	ARI	-0.04	-0.04	0	0.01	0.05	0.19
(Twitter)	AMI	-0.03	-0.03	0	0.03	0.06	0.16
Rugby	ARI	-0.03	-0.05	-0.02	-0.04	-0.01	0.17
(Twitter)	AMI	-0.02	-0.04	-0.02	-0.03	-0.01	0.04
Mean	ARI	0.096	0.093	-0.003	0.099	0.003	0.078
Across All	AMI	0.076	0.086	0.010	0.093	0.018	0.062

Table 2.18: Performance comparison between the diffusion-based, hybridized techniques and non-hybridized techniques. The hybrid techniques that have directly analogous techniques for either intermediate or late integration are compared. A negative value indicates the non-hybridized techniques performed better on the data set and performance measure than the hybridized technique.

Generally, the performance of the CNDC family of techniques relative to late integration CSPA+ or the intermediate integration CNDC_I depends on the view clusterings performance. For the CNDC_C1 technique, its performance is very similar to the intermediate integration technique of CNDC_I with some data sets like BlogCatalog and the Football data set being notable exceptions. When compared to its equivalent late integration technique, CSPA+, CNDC_C1 often performs much better across most of the data sets and only slightly worse than CSPA+ on those data sets that have a good view clustering (i.e. Twitter or Flickr). The CNDC_C2 model of-

ten performs better than CSPA+ and often slightly worse than CNDC_I. Finally, the intermediate integration technique of CNDC_I shows much better performance than CSPA+ on data sets that do not cluster well and only slightly worse performance on data sets that do cluster well. Thus, incorporation of intermediate integration information through the diffusion process generally provides better results for a subsequent late integration technique then by using the view clusterings by themselves. However, hybridization does not always result in superior performance relative to the equivalent intermediate integration technique.

In the case of the CVIC hybrid technique, it is almost uniformly better than then its late integration paradigm equivalent technique of BGPA+. The best performance gains of the CVIC technique over its equivalent late integration technique are, however, not as dramatic as the performance gains from the CNDC family of techniques compared to CSPA+. So, the integration of intermediate integration information in the CVIC model does provide for better cluster ensembling then cluster ensembling without any intermediate integration information being used.

When compared to comparable non-hybridized techniques, hybridization of multi-view clustering techniques is not always advantageous. The integration of intermediate level information into late integration techniques generally provides for performance increases, whereas the integration of late integration information in the form of view clusterings into intermediate integration techniques does not always provide for better performance. Additionally, it also seems that with hybridized techniques that there is a trade off between being robust to bad view clusterings and failing to fully utilize good view clusterings. Those techniques that use more information from the view clusterings (i.e. CNDC_C2) can have superior performance — even relative to late integration techniques — when a view clustering performs well, but also decidedly worse performance when no view clustering performs well. Overall, the empirical tests would suggest that hybridization improves late integration techniques, through the introduction of complementary information between the views to the view clusterings, but may not always improve intermediate integration techniques by the introduction of view clustering information.

2.3 Discussion

From the the empirical testing of multi-view clustering techniques on social-based data several interesting results have emerged. First, hybridization — or the incorporation of information used in different paradigms of multi-view clustering — does not always improve multi-view clustering. In general, inclusion of intermediate integration information, through diffusion with the view graphs, into the view clusterings before the final cluster ensembling nearly always produces better results and, for some data sets like the publication networks, produces much better results. The reason for this performance increase is that using just the view clusterings ignores complementary information between the views that can be used to produce better clusters. So, by diffusing the view clustering information with the intermediate level data structures of the view graphs allows for complementary information to be diffused into the view clusterings. So, it is possible to use hybridization to overcome the shortcomings of late integration techniques.

On the other hand, the inclusion of the view clusterings into intermediate integration techniques does not always produce better results and seems to be model dependent. For example in a spectral clustering model, like SFI, using the view clustering information often results in

better clusters than by just using the view graphs by themselves. This seems to be because incorporation of the view clusterings can connect what would otherwise be disconnected graphs, which are difficult for spectral clustering to cluster. In contrast to these improvements, if the intermediate clustering is done through multiplex modularity maximization, inclusion of the view clusterings almost universally results in worse clusters. Thus, it would seem that the idea of using the view clusterings does not always overcome the limitation of possible information loss by having everything mapped to the same space as is required by most intermediate integration techniques.

Those limitations aside, the use of hybridization also produced the best performances, on average, of the three paradigms across all of the data sets. This is because most of the hybridized techniques tend to be more robust across the various data scenarios. For some data scenarios intermediate integration techniques like ETL-MSC or SPSL could produce great results, and then on other data sets, produce very poor results. The same pattern was also true of the late integration techniques like DREC. Thus, while the hybridized techniques were not always the best, they were also not the worst techniques either. This robustness across data sets is a result of the greater information used in the techniques and is a promising feature of using hybridization between late and intermediate paradigm techniques in multi-view clustering.

The second main finding from the empirical testing is that there is no one clustering technique that works for all multi-view, social-based data. In fact most of the techniques that perform very well on a particular data set or data scenario often perform very poorly on other data sets or scenarios. For example, techniques like CNDC_C1 or ELT-MSC perform well on publication networks or some types of social media data respectively but then do poorly on multiplex networks or other types of social media data. This suggests first that not all social-based data scenarios are the same and so only evaluating a new technique against something like Cora or CiteSeer does not guarantee the technique's performance across the social-based data spectrum. Secondly, having some techniques perform well in some scenarios and not in others is consistent with theoretical work on clustering. Any given clustering technique must make certain trade-offs in the aspects of clustering that it attempts to fulfill [7], [76]. These trade offs often come from how the quality of how a clustering solution is defined. So, from a practical use stand-point these trade-offs in the clustering algorithm can have impacts on the types of data scenarios that a particular clustering algorithm will perform well on. For example, data scenarios that give rise to connected graphs for each view will be more amenable to spectral clustering goodness functions, as there are no disconnects in the graph that can affect spectral measures on a graph. It is important to note that this conclusion of no one clustering technique being always the best is not a limitation of the study, but an empirical confirmation of the theory and nature of clustering.

Finally, the results also suggest from a practical use standpoint that those techniques which are not always the best but also far from the worst across the various data sets (i.e. CVIC or IPMMC) may be more desirable than those that perform the best for a select group of data sets or scenarios. Since clustering is inherently an exploratory data analysis technique, the ability to produce meaningful clusters — which is itself an ill-defined concept — from new data without labels is paramount [7]. So, a technique which is stable, if not always optimal to a particular labeling-scheme for a data set, across the various social-based data sets is still of practical value. Furthermore, the technique of IPMMC which is the application of a modularity maximization technique like Leiden or Louvain to the view graphs is not only often okay across all data sets

but even occasionally gives superior performance. So, based on the results of this empirical investigation this technique will be further analyzed and developed in the next chapter.

The third main finding is that intermediate integration techniques appear to be superior to late integration techniques for multi-view, social-based data. Across nearly every data set, either intermediate integration or hybrid integration techniques produce superior results for multi-view social-based data. For those data sets on which a late integration technique had superior performance across the techniques, both intermediate and hybrid integration approaches also performed well and were very close to the same level of performance as the superior late integration technique. So, the empirical results from this study have provided further evidence that late integration techniques are generally inferior to intermediate (or hybrid) integration techniques when clustering multi-view data.

The final main finding is that the aspects of the view clusterings, and in particular having at least one view that clustered well relative to the desired labeling scheme, had a higher correlation with successful multi-view clustering than any other aspect of views. Across nearly every technique the measures relating to view clustering quality, like the best ARI/AMI of a view or the consensus between view clusterings was correlated — often strongly — with multi-view performance. In some ways, this result should be expected. After all, the more 'clusterable' a data set is, the better the performance of any suitable clustering technique should be [2]. However, in supplement to the theory for traditional clustering (i.e. with one view of the data), this 'clusterability' of the data set does not need to exist equally across all of the views; it would seem as long as at least one view is very clusterable, multi-view clustering can still be successful. It would be interesting as future research to better understand the clusterability attributes of multi-view data sets, as they seem to have more intricacies than clusterability for standard (i.e. one view) data sets.

In contrast to the positive view clustering results, only two of the graph metrics had any correlation across techniques with view performance. These metrics were the average percent of differences between the number of connected components and the number of clusters and the average clustering coefficient. In both cases, these metrics represent having a cluster structure in the graphs and so also reflect the importance of having good clusterings for the different views of a data set. What is more, these correlations to having at least one good view clustering cross all of the paradigms, whether they use view clusterings or not. This result suggests that perhaps there needs to be a third principle to multi-view clustering — derived from standard clustering theory — in addition to the *complementarity* and *consensus* principles. The third principle would be the *quality* principle which states that at least one view has the ability to cluster well. Put another way, if all views cluster poorly — even in the presence of complementary information between views and the use of intermediate integration techniques — that clustering performance will still suffer. Thus, for multi-view clustering, in addition to seeking maximum consensus between views and using the complementary information between views, one must also have at least one view that produces desirable clusters.

There are some important caveats to the findings from this chapter that need to be mentioned. First, all of the conclusions drawn in this section derive from the empirical testing of the previous section; these conclusions are only empirical conclusions and have not been definitively proven. Second, I have only focused on social-based for this work. It is not clear if any of the conclusions found extend to other types of data scenarios, like images or genetic and biological data, espe-

cially since the conclusions are founded on empirical investigations. Third, I deliberately choose a sampling of the most recent, state-of-the-art techniques in multi-view clustering. It is possible that since clustering of multi-view, social-based data has not been as thoroughly researched as other multi-view data scenarios that older techniques might actually work better on these types of data than the current state-of-the-art techniques.

2.3.1 Empirically-based Recommendations for Multi-view Clustering of Social-Based Data

Based on the findings from the testing in this chapter, I have outlined the relative strengths and weaknesses of the various techniques and provide some recommendations for usage for practitioners. The following table, Table 2.19, summarizes the strengths and weaknesses of the techniques.

Technique	Data Scenario Strengths	Performance on Strong Data Sets	Data Scenario Weaknesses	Performance on Weak Data Sets	Computational Requirements	Recommended for use on Social-Based Data?
GP-MGLA	-Twitter	Fair	-Publication Networks	Poor	High	No
CSPA+	-Twitter	Fair	-Publication Networks	Very Poor	Moderate	Only on Certain Data Scenarios
BGPA+	-Twitter -Some Social Media -Multiplex Network	Fair to Good	-Publication Networks	Poor	Low	Only on Certain Data Scenarios
LWBG	-Twitter -Some Social Media	Fair to Good	-Publication Networks -Other Social Media	Poor	Low to Moderate	No; use simpler BGPA+
MCLA+	-Twitter -Multiplex Network	Fair	-Publication Networks -Other Social Media	Poor	Low	No
LWMC	-Twitter	Fair	-Publication Networks -Other Social Media	Poor	Low to Moderate	No
DREC*	-Twitter -Multiplex Network	Very Good	-Publication Networks -Text	Poor	High	Only on Certain Data Scenarios
LPMMC	-Twitter	Good	-Publication Networks -Other Social Media	Poor	Low	No
DISC_A*	-Twitter -Some Social Media -Multiplex Network	Good to Very Good	-Publication Networks -Text	Poor	Moderate	Only on Certain Data Scenarios

DISC_M*	-Text -Some Twitter	Fair	-Publication Networks -Other Social Media	Very Poor	Moderate	No
DISC_R*	-Some Twitter -Some Social Media	Fair to Good	-Publication Networks -Text -Some Twitter	Poor to Fair	Moderate	No
DICL_A	-Twitter	Good	-Text -Multiplex Network	Fair	Moderate to High	Only on Certain Data Scenarios
DICL_M	-Some Twitter -Text -Multiplex Network	Fair to Good	-Publication Networks	Fair	Moderate to High	No
DICL_AG	-Publication Networks -Text	Good	-Twitter -Other Social Media	Fair	Moderate to High	Only on Certain Data Scenarios
DIMC_A	-Twitter	Good	-Other Social Media	Poor to Fair	Low	No; use IPMMC
DIMC_M	-Publication Networks -Twitter -Some Social Media	Fair to Good	-Other Social Media	Poor	Low	No; use IPMMC
MSIM_C	-Text	Fair	-Other Social Media -Twitter -Multiplex Networks	Poor	Moderate to High	No
CVIC	-Publication Networks -Twitter -Some Social Media	Good	-Text	Fair	Low to Moderate	Yes
CND_C1	-Publication Networks -Some Twitter -Some Social Media	Good to Very Good	-Other Social Media -Other Twitter	Poor	High	Only on Certain Data Scenarios
CND_C2	-Text -Publication Networks -Some Twitter -Some Social Media	Good	-Other Social Media -Other Twitter	Fair	High	Only on Certain Data Scenarios
CND_I	-Text -Publication Networks -Text	Good	-Other Social Media -Some Twitter	Fair	High	No; Use a Different CND model
SFI*	-Text	Fair	-Publication Networks -Other Social Media	Very Poor	Moderate	No

SPSL*	-Some Twitter	Very Good	-Publication Networks -Other Social Media	Very Poor	Moderate to High	No
CG*	-Some Social Media -Text	Good	-Some Twitter	Fair	High	No
ResK*	-Multiplex Network	Fair	-Publication Networks -Other Social Media -Twitter	Very Poor	High	No
IPMMC	-Other Social Media -Text -Some Twitter	Good to Very Good	-Some Twitter	Fair	Low	Yes
NF-CCE*	-Other Social Media -Multiplex Network	Fair to Good	-Twitter	Poor to Fair	High	No
ETL-MSC*	-Publication Networks -Other Social Media	Very Good	-Twitter -Multiplex Network	Very Poor to Poor	High	Only on Certain Data Scenarios

Table 2.19: Summary of the performance and recommend usage situations for the various multi-view clustering techniques tested in this chapter. The data scenarios where techniques perform well and poor at, along with how good or poor their performance is when clustering in that data scenario are summarized. I have also provided a brief comparison of the computational requirements for the different techniques and their recommended usage. Only IPMMC and CVIC are recommended for usage generally.

In terms of practical use of multi-view clustering algorithms on social based-data, there are some steps which should be observed.

1. Ensure that one of the views has a good clustering of the data. There is no specific rule as to what constitutes a good clustering, as different types of data and different algorithms have different measures of quality (and there are no 'ground truth' labels as there are in the benchmark data sets). From the empirical tests in this chapter I have found measuring either the modularity of the clustering of the network modes or view graphs or the average clustering coefficient of the network modes or the view graphs to be an indicator of a view's clustering performance. If no view has good clustering performance, it is likely multi-view clustering will also not recover good clusters and more or different data may be needed.
2. Compare the data scenarios to those listed in Table 2.3 and Table 2.3. If the data scenario closely resembles one of those, then use the techniques that perform best on that particular data scenario, from Table 2.19. Care should also be taken to pick a technique that can handle the size of the data; there is a wide variance in computational demands by the techniques. For example, if the objective is to find clusters in a corpus of texts and citations of a few thousand in number, then something like CNDC_C1 should be the technique. Whereas, if the objective is to find communities in a large sample of Twitter data, then something like IPMMC or possibly BGPA+ should be used.
3. If the data scenario does not match any of those explored, then something either CVIC or IPMMC should be used. While these two techniques may not give the best clusters, they are robust across the data sets and will be mostly likely to produce reasonable cluster assignments.

In summary, this chapter has proposed a new hybrid paradigm to clustering multi-view data. I then tested the new hybrid paradigm techniques along with several state-of-the-art intermediate integration and late integration techniques using social-based, multi-view data. These results of these tests indicated that hybridization can sometimes help multi-view clustering, especially when compared to late integration techniques. Also, multi-view, social-based data poses distinct challenges for multi-view clustering due to the wide variety of data types and interactions. As a result, no particular multi-view clustering technique is superior across the range of multi-view, social-based data. However, some techniques can give superior performance in certain types of social-based data and two techniques showed robust performance across the range of social-based data. These results indicate that more research should be done into those techniques which were more stable in their ability to produce meaningful clusters from data.

Chapter 3

Network Modularity-based Clustering of Multi-View Data

In this chapter I will investigate the use of modularity and converting all views of a multi-view data set to graphs as a means of finding clusters in the data. In particular, I investigate modularity's suitability as an optimization objective when clustering multi-view, social-based data. I then present a new algorithm that further improves the use of modularity for multi-view clustering, MVMC, which is able to automatically adjust parameters like the view weights and view resolutions. I then demonstrate the algorithms effectiveness in clustering multi-view, social-based data using a collection of benchmark, multi-view data sets that come from different social-based data generation scenarios. The results of the empirical testing not only demonstrate the effectiveness of the proposed method, but also show that the proposed method has certain desirable properties that give a user useful, evidence-based measures to judge the clusters produced by the algorithm and the clusterability of the data set.

the organization of the chapter is as follows. In the first section, I present some salient background information on modularity and its use in clustering. I then perform empirical testing to demonstrate the suitability of modularity for multi-view clustering. Then, I describe a new modularity-based algorithm, along with some variants, to use for clustering multi-view, social-based data. Finally, I empirically test the new algorithm against other state-of-the-art algorithms, which were used in the previous chapter, across a collection of benchmark data sets to demonstrate the proposed methods effectiveness.

3.1 Background

Network modularity is one of the most frequently used functions to describe communities within real-world networks. As such, it is often used as the objective function in network clustering techniques [94], [13]. Generally, network modularity measures the amount of links internal to a cluster versus what would be expected to exist if links were placed at random. It has the form of:

$$Q = \frac{1}{2|E|} \sum_{ij \in E} [A_{ij} - P_{ij}] \delta(C_i, C_j) \tag{3.1}$$

where the δ function is one if nodes i and j are in the same cluster and $|E|$ are the number of edges present in the graph (note, this can be substituted with sum of all the edges in the graph, $\sum E$ in a weighted graph scenario). In terms of the null-model used for determining the significance of the links, P_{ij}, the most common choice is:

$$P_{ij} = \frac{deg(i) \times deg(j)}{2|E|} \tag{3.2}$$

for undirected networks and:

$$P_{ij} = \frac{deg_{in}(i) \times deg_{out}(j)}{|E|} \tag{3.3}$$

for directed networks, where $deg(i)$, $deg_{in}(i)$, and $deg_{out}(i)$ are the total degree, in-degree, and out-degree for node i [94], [44]. Despite its empirical successes and strong grounding in statistical physics, modularity does have an important shortcoming: modularity has a *resolution* limit. In brief, the resolution limit is when modularity cannot detect clusters present in a network when the network is sufficiently large and communities are sufficiently small [80], [50]. In order to address this shortcoming, several authors have proposed various means of correcting for different resolutions when clustering networks [127], [38], [108], [128]. One means of addressing the resolution limit is to add a parameter to the modularity to directly account for the resolution that may be present in a network:

$$Q = \frac{1}{2|E|} \sum_{ij \in E} [A_{ij} - \gamma \frac{deg(i) \times deg(j)}{2|E|}] \delta(C_i, C_j) \tag{3.4}$$

where γ is now a resolution parameter that can be set depending on the number of clusters that are present in the network relative to the network's size [108]. Another way of address the resolution limit is to use 'resolution-free' definitions of modularity [127], [38]. Typically, these definitions of modularity include a new term relating the network's structure to the internal density of communities, rather than a null-model. A prominent example of these formulations is the Constant Potts Model:

$$Q = \frac{1}{2|E|} \sum_{ij \in E} [A_{ij} - \gamma] \delta(C_i, C_j) \tag{3.5}$$

where now the resolution parameter of γ is a boundary between the internal and external densities of non-isolate communities in a network [127].

Due to modularity's widespread success as an optimization objective for clustering networks, it has been subsequently adapted to cluster both attributed networks and multi-layer networks [32], [93]. for the general case of multi-layer networks, network modularity can be broken into two components of inter-layer modularity plus intra-layer modularity:

$$\begin{aligned} Q &= \sum_{v=1}^{m} \sum_{ij \in E^v} [A_{ij}^v - \gamma^v \frac{deg(i)^v \times deg(j)^v}{2|E^v|}] \delta(C_i^v, C_j^v) + \sum_{v=1}^{m} \sum_{s \neq v} \sum_{i=1}^{n} \omega_i^{sv} \delta(C_i^s, C_j^v) \\ Q &= Q_{intra} + Q_{inter} \end{aligned} \tag{3.6}$$

for multiplex clustering, or multi-view clustering, where clusters are not allowed to vary between views or layers, this modularity can be reduced to just the sum of the intra-layer modularities. Furthermore, it is also possible to weight the different views, or layers, differently as well. So, a weighted multi-view modularity would have the form of:

$$Q = \sum_{v=1}^{m} w^v \sum_{ij \in E^v} [A_{ij}^v - \gamma^v \frac{deg(i)^v \times deg(j)^v}{2|E^v|}] \delta(C_i, C_j) \tag{3.7}$$

where w^v is the weight of a particular view. It was this form of modularity that was used in the previous chapter in the IPMMC, LPMMC, DIMC_A, and DIMC_M, and generally showed good empirical performance. However, in these methods the view resolutions and weights were all set to one. This formulation does, however, present the additional problem of how to determine appropriate view weights.

The weighting of different views in a multi-view clustering quality function is not new. It has seen use in techniques which use spectral clustering [150], [9], [10]. In this case the weighting term can be updated in an iterative fashion by using something like the eigenvectors, X, of the weighted combination of graphs to update the weights by measuring the smoothness of the eigenvectors through each graph's Laplacian by $tr(X^T L^v X)$, and down-weighting those views which are not smooth. The problem with this form of weighting is that a graph with disconnects will naturally be very un-smooth and so will be down-weighted even if it contains useful information for the cluster structure. So, selecting appropriate view weights in an adaptive and automatic way is still an open problem, and dependent on the nature of the cluster goodness function.

3.2 Use of Modularity in Multi-view Clustering

In this section I will investigate the modularity function's appropriateness for use in multi-view clustering of social-based data. As has been noted earlier, the IPMMC method, which produced the most stable and some state-of-the-art clusterings on benchmark social-based data sets, uses network modularity as the function being maximized to determine a good clustering of the data. Given this reliance on the measure of modularity for the clustering of the data, it is important to know how well the function works as an objective function for clustering multi-view, social-based data.

For these experiments, the modularity being maximized is the unweighted sum of the modularities from all of the view graphs. The function is as follows:

$$Q = \sum_{v=1}^{m} \frac{1}{2|E|} \sum_{ij \in E^v} [A_{ij}^v - \frac{deg(i)^v \times deg(j)^v}{2|E|^v}] \delta(C_i, C_j) \tag{3.8}$$

So, the summation across the views uses each view's adjacency matrix, A^v, but only has one clustering across all of the views C. It should be noted that if a particular view is a directed graph, one can easily substitute in the modularity for a directed graph.

In order to evaluate the modularity function, I looked at four measures across all twelve of the benchmark data sets for all of those methods that optimize on network modularity from the previous chapter (IPMMC, LPMMC, DIMC_A, and DIMC_M). The first two measures are

the Pearson correlation coefficient of the ARI/AMI of the clustering and the final, optimized modularity. This is to determine if when the techniques obtain a good modularity value, is the clustering also good and vice versa. Finding the optimal modularity value in the technique should correspond to finding a good clustering of the data. The third measure is the correlation between modularity and a cluster goodness indicator (i.e. ARI) over the optimization procedure. This measures if when the optimization function improves, does the clustering quality also improve. The final evaluation measure is the difference between the modularity value of the benchmark clustering and the modularity of the technique's optimal modularity. This will give an indication of the appropriateness of the model in terms of capturing the desired, benchmark labels. The optimal modularity found by the technique for a given graph should be close to the modularity that would be produced on the same graph by the benchmark clusterings. The following table, Table 3.1, displays these measures for the modularity-based techniques of IPMMC, LPMMC, DIMC_A, and DIMC_M. It should be noted that the primary difference between these techniques is the graph being used; IPMMC just uses graphs of the data while LPMMC uses the cluster association graphs, and DIMC_A, and DIMC_M use combinations of the latter. For the final two measures, only the averages across all of the data sets are reported here. The full values of the measures both across the data sets and the views of the data sets are available in the Appendices (Appendix A).

Comparison Measure	IPMMC	LPMMC	DIMC_A	DIMC_M
Correlation between modularity and ARI	0.318	0.649	0.456	0.418
Correlation between modularity and AMI	0.529	0.767	0.451	0.594
Avg. correlation of modularity improvement over optimization	0.984	0.995	0.773	0.984
Avg. of absolute difference between modularities of found clusters and benchmark clusters	0.193	0.122	0.130	0.101

Table 3.1: Correlation between the clustering qualities and the modularities of the various modularity-based multi-view clustering methods. While all of the methods do have positive correlation, some methods, which primarily differ in the graphs use, have a stronger correlation to the clustering quality.

Based on the testing of the modularity function, two primary results supporting its use in multi-view clustering emerge. First, for each technique there is a positive correlation between the resultant modularity value and the actual quality of the resultant clusters, relative to the benchmark data sets. The IPMMC method, despite being the best performing method of all of the four, does have the lowest correlation at 0.318. For some of the data sets with more sparse graphs (i.e. Cora, AUCS, etc.), the technique finds higher modularity clusters than what would be found from the benchmark clusterings. This result is further supported by the differences between the modularity produced by the clustering from the techniques versus the modularity produced by the benchmark clusterings. The IPMMC method has the highest difference between the two modularity values on average. Second, for each model, improvement in modularity highly correlates with improvement in the cluster quality over the course of the optimization.

That is to say that as a better modularity value is sought, a better clustering solution emerges.

Taken all together, these results do not at first glance strongly support the use of modularity as the optimization function for multi-view clustering of social-based data. While improvements in modularity do correlate strongly with improvements in the clustering, the final modularity only moderately correlates with the actual quality of the clustering, relative to the benchmark labels. However, this correlation between the final modularities and the clustering performance also seems to be highly model dependent. IPMMC has a much lower correlation than does LPMMC with the other two techniques having values in between. Since the primary difference between these techniques are the graphs of the views that are being used, this would suggest that if more appropriate view graphs that better encode the cluster structure were used, that the clusterings would be better. However, it should be noted that there are also performance differences between these techniques as well, with IPMMC actually having the best performance and LPMMC having worse performance. So, while optimizing on modularity does seem to produce good results and seems to be a reasonable optimization function to use in multi-view clustering of social-based data, it's quality is dependent on how the views are modeled as graphs and a good modularity is not a guarantee of a good clustering.

3.3 Multi-view Modularity-based Clustering Methods

In this section I will outline two new techniques that modify the technique of Intermediate Paradigm Modularity Multi-view Clustering (IPMMC) that was proposed in the preceding chapter. The IPMMC method uses standard network modularity, which is known to have some issues with being an objective function used in clustering graphs, and considers all views as equal in terms of their contributions to the multi-view clusters. As was described in the introduction to this chapter, one of the chief issues with modularity is its resolution limit. Two means of overcoming this resolution limit is to adopt a resolution parameter into the modularity function or to use one of the 'resolution free' density-based modularity functions. In the case of using a resolution parameter in the modularity function, recent work has shown what the optimal value of that resolution parameter should be for a given graph, given some assumptions about the nature of the graph [95]. This work has been even more recently extended to multi-layer graphs and been used to define what the optimal weight for each layer should be when using a modularity function with a resolution parameter [101]. Given these means of correcting for the deficiencies in the modularity function, this section will use these corrections within the same procedure as IPMMC to produce a better means of clustering multi-view, social-based data.

Common to both the proposed methods in this section and IPMMC is the framework of clustering multi-view data. In all of these techniques the general framework is to convert each view to a graph, if it is not a graph already, and then simultaneously cluster the graphs using modularity as the optimization function for the clustering. It should be noted that many of the multi-view clustering techniques employed in multi-view image clustering and multi-view clustering of medical data use the same first step of transforming all of the views into graphs [148], [147], [107]. However, none of these techniques, to the best of the author's knowledge, ever employ network modularity as the optimization function when clustering the multi-view networks. Within the realm of social-based data, only one technique in attributed graph clustering does a

similar first step of converting the attributes to graphs [67]. The paper only considers a limited case of attributes in that there is only one view of categorical attributes. So, the common framework used by IPMMC and the two proposed methods here is new to social-based, multi-view data. The following figure, Figure 3.1, displays the overall framework.

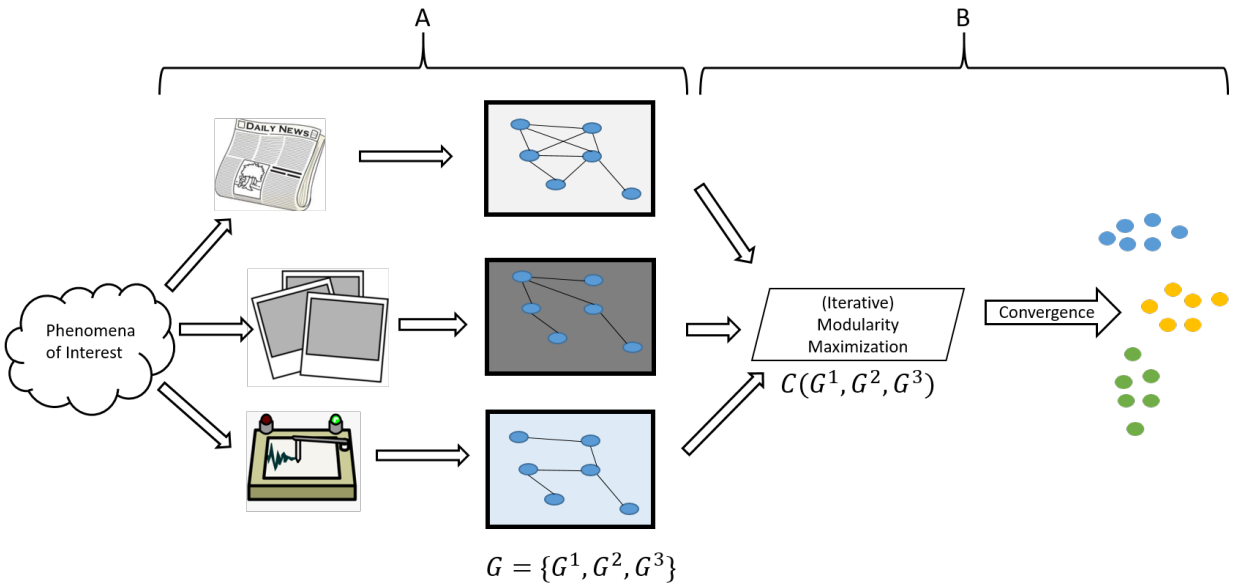


Figure 3.1: Overview of the Multi-view Modularity Clustering method. the method works in two main steps. In step A, all of the views of the data are converted into graphs. In step B, these view graphs are collectively clustered using an iterative modularity maximization technique to produce clusters.

The methodology proposed in this work for clustering multi-view. social-based data consists of two steps. The first step is to form graphs for every view of the data (A in Figure 3.1). This has been done by the computationally quick procedure of using a symmetric k-Nearest Neighbor graph, where k is the square root of the number of vertices, $k = \lfloor \sqrt{n} \rfloor$, for modes that are not already networks or graphs [106]. While this method can be used for creating the graphs, there are many other methods for inferring or learning graphs from data which could easily be used in this methodology [23], [106]. Transforming the data into a graph space provides two important advantages for multi-view clustering. First, it allows for the definition of a clustering goodness function that is parsimonious to all combinations of views of social-based data. Second, graphs better preserve the latent manifold structure of the data which allows for better structure, like clusters, to emerge [106]. Having formed graphs of all of the views, the next step is to use a modularity-based clustering procedure to simultaneously cluster all of the views (B in Figure 3.1). Based on the results of the previous chapter, modularity-based clustering — especially in an intermediate integration paradigm — gives some of the best clustering results. Having established the overall methodology, I will now describe in detail the means of performing modularity-based clustering.

3.3.1 Multi-view Modularity Clustering

For the first proposed technique, the weighted and resolution-corrected network modularity function will form the clustering quality function. that is to say, the quality of any clustering over the multi-view data is evaluated by:

$$Q = \sum_{v=1}^{m} w^v \sum_{ij \in E^v} [A_{ij}^v - \gamma^v \frac{deg(i)^v \times deg(j)^v}{2|E^v|}] \delta(C_i, C_j) \tag{3.9}$$

In effect, clustering with this function is optimizing Q, given the appropriate weight and resolution parameters. However, since the weight and resolution parameters are not always know *a priori* this function can be considered as having three, free parameters for determining the quality any given clustering: the cluster assignments, C, the view weights, w^v, and the view resolutions, γ^v. So, the clustering technique can be written as optimizing the following problem:

$$\begin{aligned} \max_{w,C,\gamma} \quad & \sum_{v=1}^{m} w^v \sum_{ij \in E^v} [A_{ij}^v - \gamma^v \frac{deg(i)^v \times deg(j)^v}{2|E^v|}] \delta(C_i, C_j) \\ s.t. \quad & C \text{ is integer} \\ & \gamma, w \in \mathbb{R} \end{aligned} \tag{3.10}$$

In order to solve this problem, I have adopted an alternating, iterative optimization procedure. In the first step, γ and w are fixed and C is adjusted by a modularity optimization procedure. In the second step, C is fixed and γ and w are adjusted. This alternating iterative procedure performs well in practice at finding good solutions and has certain other benefits for evaluating the quality of solutions, which will be seen later in the section on empirical testing.

For the first step of the procedure, a standard modularity maximizing clustering procedure (i.e. Louvain, Newman-Girvan, Leiden, etc.) can be used to find the the best values for C, given the resolutions and weights remain fixed. This then leaves the problem of how to optimally set the resolutions and weights, once the cluster assignments are fixed. To do so, I utilize analytically derived equations for producing the optimal values for the weights and resolutions, given a clustering. In order to set the the resolution parameter to an optimal value for a particular graph, the following function is used:

$$\gamma = \frac{\theta_{in} - \theta_{out}}{\log\theta_{in} - \log\theta_{out}} \tag{3.11}$$

where γ is the resolution parameter, and θ_{in} and θ_{out} are the propensities of having edges internal to clusters or external to clusters respectively. This function was derived by relating modularity maximization to the planted partition Stochastic Block Model [95], [101]. The intuition behind this function is that when there is a greater propensity to form edges internal rather than external to clusters that the resolution should be higher which would bias the function to find more tightly-knit and possibly smaller communities. Since this formulation relies on knowing the clusters to compute the θ values, it does not on first glance seem useful for actually clustering a graph. However, the function for computing the resolution has been used in an iterative fashion with modularity-based graph clustering to optimally cluster graphs in reasonably few (i.e.

less than 20) iterations [95], [101]. So, it is possible to iteratively cluster and then update the resolution parameter to both find the optimal clustering of the graph as well as its appropriate resolution parameter.

The same means of computing the optimal resolution parameter can be extended to compute the optimal view weights. Pamfil et al. used the same derivation process of finding the optimal resolution parameters from the relation of Reichardt and Bornholdt modularity to planted partition Stochastic Block Models to obtain the optimal weights for each of the layers in a multiplex graph as:

$$w_v = \frac{\log\theta_{in}^v - \log\theta_{out}^v}{< \log\theta_{in}^v - \log\theta_{out}^v >_v} \tag{3.12}$$

where w_v is the weight given to a view, v, and θ_{in}^v θ_{out}^v are the propensities for edges to form internal to a cluster or external for the vth view, respectively. $< . >_v$ is the average across all of the views. The intuition behind this derivation is that those views with higher propensity to have edges internal to clusters versus having edges external to clusters relative to the average across all views will have higher weights. So, a view with a better than average propensity to have edges internal to clusters should be weighted more heavily in the modularity calculation for clustering. It should be noted that this version of choosing the weights does not suffer from the same disconnected graph problem that the spectral version of updating the weights does. Once again, as with the resolution parameter, the weight parameter can be determined in an iterative fashion [101].

Having defined the new objective function as well as means of optimizing that function, a new algorithm can then be developed to cluster multi-view data. At a high level, the algorithm runs by first assigning starting resolution and weight parameters for every view (typically one) and then clustering the graph using a modularity maximization technique like Louvain or Leiden. These clusters are then used to compute new resolution and weight parameters. This process is repeated until the resolution and weight parameters no longer change. In the event that the resolution and weight parameters do not converge (which can happen in practice [101]), the clustering with the highest modularity value is chosen as the final clustering. The following psuedocode, Algorithm 9 describes the algorithm in detail.

Algorithm 9 Multi-view Modularity Clustering (MVMC)

input:

- Adjacency for each view: A^v
- Max number of iterations: $max\_iter = 20$
- Starting resolutions: $\gamma_1^v = 1, \forall v \in m$
- Starting weights: $w_1^v = 1, \forall v \in m$
- Convergence tolerance: $tol = 0.01$

output: Cluster assignments

$clustering^* \leftarrow None$

$modularity^* \leftarrow -\infty$

for $i = 1 : max\_iter$ **do**

$clustering_i \leftarrow cluster(A, w_i, \gamma_i)$

$modularity_i \leftarrow RBmodularity(A, clustering_i, w_i, \gamma_i)$

$\theta_{in}, \theta_{out} \leftarrow calculate\_thetas(A, clustering_i)$

$\gamma_{i+1}^v \leftarrow \frac{\theta_{in}^v - \theta_{out}^v}{\log\theta_{in}^v - \log\theta_{out}^v}, \forall v \in m$

$w_{i+1}^v \leftarrow \frac{\log\theta_{in}^v - \log\theta_{out}^v}{<\log\theta_{in}^v - \log\theta_{out}^v>_v}, \forall v \in m$

if $abs(\gamma_{i+1} - \gamma_i) < tol$ AND $abs(weights_{i+1} - weights_i) < tol$ **then**

$clustering^* \leftarrow clustering_i$

$modularity^* \leftarrow modularity_i$

BREAK

end if

if $iter >= max\_iter$ **then**

$best\_iteration \leftarrow argmax(modularity)$

$clustering^* \leftarrow clustering[best\_iteration]$

$modularity^* \leftarrow modularity[best\_iteration]$

end if

end for

return $clustering^*$

The algorithm begins by initializing all the resolution parameters, γ_1^v, and weight parameters, w_1^v to one (or whatever the user may specify). The algorithm then goes on to cluster the view graphs, A^v, by a modularity maximization technique (i.e. Louvain, Leiden), $cluster()$, with the current resolution and weight settings. The output of this is then used to determine the propensities for internal edge formation θ_{in}^v, and external edge formation, θ_{out}^v for each view. These values are then used to update the resolution, γ^v, and weight parameters, w^v, for each of the views. If the new weight and resolution parameters are the same as the previous ones (within tolerance), the algorithm then exists and returns the final clustering. If the algorithm fails to converge to stable resolution and weight parameters, within the maximum number of iterations allowed, then the algorithm returns whichever clustering produced the highest modularity. Note, that modularity for this algorithm is the view-weighted, Reichardt and Bornholdt modularity, which incorporates the view modularities.

One of the important elements in the aforementioned algorithm, Algorithm 9, is the computation of the edge propensities, θ. In order to calculate these edge propensities, I follow the

guidance outlined in previous works and assume edges form by a degree-corrected model [95], [101]. Given a degree corrected model, the expected number of edges that occur internal to clusters is given by:

$$\begin{aligned} e_{in} &= \frac{1}{2}\sum_{c}\sum_{ij\in E}\theta_{in}\frac{deg(i)deg(j)}{2|E|}\delta(C_i,C_c)\delta(C_j,C_c) \\ &= \frac{\theta_{in}}{4|E|}\sum_{c}\kappa_c^2 \end{aligned} \tag{3.13}$$

where c is a cluster and $\kappa_c = \sum_i deg(i)\delta(C_i, C_c)$, or the sum of the degree of the vertices within cluster c. Using the observed number of edges internal to the clusters for the expected number of edges internal to clusters, e_{in}, this equation can then be used to calculate the propensity to form edges internally as:

$$\theta_{in} = \frac{e_{in}}{\sum_c \frac{\kappa_c^2}{4|E|}} \tag{3.14}$$

The same update can be derived for directed graphs as:

$$\theta_{in} = \frac{e_{in}}{\sum_c \frac{\sum_{ij\in c} deg_{in}(i)deg_{out}(j)}{2|E|}} \tag{3.15}$$

Similar to the propensity to form edges internally, the propensity to form edges externally can be derived from the expected external edges under the degree-corrected model as:

$$\theta_{out} = \frac{|E| - e_{in}}{|E| - \frac{\sum_c \kappa_c^2}{4|E|}} \tag{3.16}$$

and for directed graphs as:

$$\theta_{out} = \frac{|E| - e_{in}}{|E| - \frac{\sum_c \sum_{ij\in c} deg_{in}(i)deg_{out}(j)}{2*|E|}} \tag{3.17}$$

With these equations, and the assumption of edges forming by a degree-corrected model, the propensities for edges to occur internal or external to a cluster can be calculated. It is important to note, however, that these equations assume the *observed* edges internal or external to the clusters are equal to the *expected* edges internal or external to clusters. In practice, these values may actually differ. For example, if every vertex ends up in its own cluster, than no edges will be internal which will lead to the propensity internal term, θ_{in} to become zero, and the resolution and weight updates to fail. A similar problem can occur if all the vertices end up in one cluster and so no external edges occur. While at first glance these examples may appear to be edge cases, these problems can also occur in more important cases. For example, if the graph consists of a series of disconnected cliques. The optimal clusters in this situation would then be to put all of the cliques within their own, separate clusters. However, this would result in there being no external edges, and so the resolution and weight updates would fail. In order to address these

shortcomings, I have chosen to have a small value substitute for the propensities if there are either no internal edges and/or no external edges. So, while the observed edges can act as a proxy for the expected edges in most cases, the expected edges will always be a nonzero number. With these corrections, the algorithm for computing the propensity values is given in pseudocode by Algorithm 10.

Algorithm 10 Calculation of Edge Propensities

input:
- Adjacency for each view: A^v
- clustering: C

output: Internal and external edge propensities (θ_{in}, θ_out)
for v = 1:m **do**
  $e_{in} = 0$
  $\kappa^2 = []$
  for c = 1:—C— **do**
    $e_c = |E_c^v|$
    $e_{in} += e_c$
    if A^v is directed **then**
      $\kappa^2.append(\sum_{ij \in V_c^v} deg_{in}(i) deg_{out}(j))$
    else
      $\kappa^2.append((\sum_{i \in V_c^v} deg(i))^2)$
    end if
  end for
  if A^v is directed **then**
    if $e_{in} = 0$ **then**
      $\theta_{in}^v \leftarrow \frac{1}{|E^v|}$
    else
      $\theta_{in}^v \leftarrow \frac{e_{in}}{\sum \frac{\kappa^2}{2|E^v|}}$
    end if
    if $e_{in} == |E^v|$ **then**
      $\theta_{out}^v \leftarrow \frac{1}{|E^v|}$
    else
      $\theta_{out}^v \leftarrow \frac{|E^v| - e_{in}}{|E^v| - \sum \frac{\kappa^2}{2|E^v|}}$
    end if

else
if $e_{in} = 0$ **then**
$\theta^v_{in} \leftarrow \frac{1}{|E^v|}$
else
$\theta^v_{in} \leftarrow \frac{e_{in}}{\sum \frac{\kappa^2}{4|E^v|}}$
end if
if $e_{in} == |E^v|$ **then**
$\theta^v_{out} \leftarrow \frac{1}{|E^v|}$
else
$\theta^v_{out} \leftarrow \frac{|E^v| - e_{in}}{|E^v| - \sum \frac{\kappa^2}{4|E^v|}}$
end if
end if
end for
return $\theta_{in}, \theta_{out}$

The algorithm goes through each graph to calculate the propensities for each graph separately. For each graph, the algorithm begins by calculating the number of internal edges and the degree-corrected, null-model terms (i.e. κ^2)for each of the clusters. Then, the algorithm checks as to whether the graph is directed or undirected and whether there are no internal or external edges and then calculates the final propensities for that view graph, $\theta^v_{in}, \theta^v_{out}$. Once the propensities have been calculated for all of the view graphs, these are then returned.

3.3.2 Multi-view Density-Modularity Clustering

As was mentioned in the introduction to this chapter, another means of addressing the resolution problem is to use one of the density-based modularities. In particular, the Constant Potts Model has been proven to have very desirable resolution-free properties [127]. Despite these favorable properties and in contrast to the Reichardt and Bornholdt modularity there is no known way to appropriately set the resolution parameter, γ, in the absence of some ground-truth knowledge about the communities. All of the density-based modularity works use *a priori* knowledge about the probabilities of edges forming internal and external to clusters to set the resolution parameters [127], [38]. Furthermore, the resolution parameter in these modularities acts as a bound on the internal and external edge densities of the clusters present within the graph. So, at first glance it would seem the optimal way of setting the resolution parameter would be to set it to a value that maximizes the internal edge density of the clusters under the intuition that denser clusters are better clusters. Unfortunately, this has the effect of causing the optimization procedure to over-cluster the graph, which produces many, small — but internally dense — clusters which are generally not meaningful. In other words, for density-based modularities there exists a balance between the internal density of clusters and the number of clusters when it comes to finding the optimal resolution parameter.

In the course of this work I have tried several ratios that attempt to balance internal cluster density with the number of clusters but did not find any that provided good performance in accurately determining the resolution parameter without ground-truth knowledge of the clusters

that are sought. Additionally, trying to update the weights of the different views by the same intuition as was done with the MVMC method (i.e. those views which have a higher than average difference between internal and external edge propensities should be weighted more heavily) did not produce significant differences in the outcomes and so a weight update step is omitted in this technique. Since there is no good means of determining the resolution parameter of density-based modularities, I have instead adopted a grid search framework over a set of possible resolution values, and selecting that resolution (and clustering) which gave the highest modularity. The following psuedocode, Algorithm 11, displays the algorithm for multi-view density modularity clustering.

Algorithm 11 Multi-view Density Modularity Clustering (MVDMC)

input:
- Adjacency for each view: A^v
- Resolutions: $\gamma_{poss} = [0.0001, 0.001, 0.005, 0.01, 0.05]$
- Weights: $w^v = 1, \forall v \in m$

output: Cluster assignments

$clustering^* \leftarrow None$
$modularity^* \leftarrow -\infty$
$\gamma = cross\_product(\gamma_{poss}, m)$
for $i = 1 : |\gamma|$ **do**
  $clustering_i \leftarrow cluster(A, w, \gamma_i)$
  $modularity_i \leftarrow modularity(A, clustering_i, w)$
  if $modularity_i > modularity^*$ **then**
    $clustering^* \leftarrow clustering_i$
    $modularity^* \leftarrow modularity_i$
  end if
end for
return $clustering^*$

The algorithm first sets the best modularity found to $-\infty$ in order to make sure at least one modularity value will be selected form its iterations, and creates an array of resolutions using the cross product such that every combination of view and possible resolutions γ_{poss} is tried. These different combinations of views and resolutions are necessary as there are different graph topologies between the views and so having one resolution value for across all views will likely produce worse results. The algorithm then uses these view resolutions to perform modularity maximization with the Constant Potts Model of resolution. If the clustering produced at those resolution values is better than the current best clustering, than it becomes the new best clustering, $clustering^*$. It is important to note that standard modularity is used to judge the clusterings, and not the Constant Potts Model modularity that was used in the optimization. In the course of testing, it was observed that picking the clustering that had highest Constant Potts Model modularity was never the best clustering, in large part due to the density issues described previously. It is also important to note that the number of runs of the algorithm increases exponentially with the inclusion of more possible modularities or more views of the data. Thus, given all of the assumptions and alterations needed to form this algorithm, this algorithm is not intended for practical

clustering use, but rather used to investigate the suitability of resolution-free, density-based modularities for clustering multi-view, social-based data within the proposed methodology.

3.4 Empirical Testing

In this section I evaluate the proposed algorithm, multi-view modularity clustering (MVMC), including investigating the use of density-based modularities (MVDMC) using benchmark, multi-view, social-based data sets. The data sets used for evaluation are the same twelve benchmark data sets from the previous chapter. These data sets encompass various types of social-based data scenarios, like social media and publication networks. For each data set, each method was run 20 times and the averages across the 20 iterations are reported. In order to evaluate the quality of the clusters found by different techniques, I once again used the Adjusted Rand Index (ARI) and Adjusted Mutual Information (AMI) [69], [132].

The settings for the algorithms used in the tests are the same as those from the previous chapter. The MVMC algorithm was allowed to run for a maximum of 20 iterations, and the resolutions for all of the views were initialized to one. Two different settings for the initial values of the weights were tried: all views set to one and each view being weighted by its sparsity divided by the maximum sparsity across all of the modes. The reasoning behind trying the second initialization was to see first, if changing the initialization changes the outcomes, and second because the denser view graphs are generally (but not always) the less useful graphs for cluster determination. So, the sparsest view graph will have am initial weight of one, and the other view graphs will have weights less than one. Finally, the tolerance for determining whether the resolutions and weights have converged was set to 0.01. For the MVDMC variant the possible resolution values tried were: $\gamma_{poss} = [0.0001, 0.001, 0.005, 0.01, 0.05]$. These values were determined by some initial trials of MVDMC on the data set and seemed to encompass the range of resolution values that would give good clusters.

3.4.1 Multi-view Modularity Clustering Results

In this section I evaluate the clustering performance of the MVMC algorithm. First, I compare whether the proposed algorithm of MVMC, which modifies the IPMMC algorithm, actually performs better than the IPMMC algorithm. The following table, Table 3.2, summarizes the difference in performance between MVMC and its simplified, progenitor algorithm of IPMMC.

Data set	ARI	AMI	Difference with IPMMC ARI	Difference with IPMMC AMI
Cora	0.36	0.44	0.03	0.09
CiteSeer	0.39	0.38	0.02	0.04
Flickr	0.63	0.61	0.06	0.01
BlogCatalog	0.64	0.62	0.01	0.09
Wiki	0.294	0.498	0.004	0
3Sources	0.73	0.72	-0.03	-0.02
AUCS	0.632	0.739	0.062	0.049
Football	0.87	0.92	0.22	0.09

Olympics	0.73	0.87	0.19	0.1
Politics Ire	0.914	0.88	0.024	0.02
Politics UK	0.99	0.97	0	0
Rugby	0.64	0.68	-0.01	0

Table 3.2: Performance of MVMC and its progenitor algorithm IPMMC on their ability to produce good clusters relative to the benchmark clusters. In every case, the MVMC algorithm performs as good or better than IPMMC, indicating that setting appropriate resolution and weight parameters provides a distinct benefit. The data sets where there are slight negative values in performance for MVMC are a result of random fluctuation and not indicative of worse performance than IPMMC.

Across all of the data sets the MVMC method performs as good or better than the IPMMC method. While there are some small negative values in the performance comparison for MVMC, these small values are the result of the random fluctuations of the algorithms and not a result of generally worse performance. The small negative and positive values for the differences in performance between the two methods indicate the same performance. The most notable improvements of the MVMC method over the IPMMC method are for the Football and Olympics twitter data sets. It would seem that even with smaller data sets there can be not only differences between the views but also significant resolution effects. These results demonstrate both the superiority of the MVMC method over its simpler counterpart and emphasize the importance of having appropriate resolution values for different data sets, which has been observed in other works [95]. Having established MVMC's performance relative to its simplified version of IPMMC, I now turn to how the algorithm does against all of the other algorithms considered in this work. The following table, Table 3.3 summarizes MVMC's performance as compared to the best algorithm for each data set.

Data set	ARI	Difference with Best ARI	AMI	Difference with Best AMI	Avg. number of clusters	Difference in number of clusters	Best Method
Cora	0.36	-0.08	0.44	-0.05	10.25	3.25	CNDC_C1
CiteSeer	0.39	-0.04	0.38	-0.03	8	2	CNDC_C1
Flickr	**0.63**	**0**	0.61	-0.04	19.05	10.05	ETL-MSC
BlogCatalog	0.64	-0.1	0.62	-0.09	9	3	ETL-MSC
Wiki	0.294	-0.046	**0.498**	**0.008**	18.9	1.9	ETL-MSC
3Sources	0.73	-0.03	0.72	-0.02	4.65	-1.35	IPMMC
AUCS	0.632	-0.038	0.739	-0.061	6	-3	DREC
Football	**0.87**	**0.04**	**0.92**	**0.03**	22.9	2.9	DISC_A
Olympics	0.73	-0.17	0.87	-0.07	45.9	17.9	DREC
Politics Ire	0.914	-0.006	**0.88**	**0.01**	4.3	-2.7	DICL_M
Politics UK	**0.99**	**0**	**0.97**	**0**	4	-1	IPMMC
Rugby	0.64	-0.08	0.68	-0.02	10	-5	CMIC

Table 3.3: Summary of the performance comparison to the clusters produced by MVMC and those produced by the method that performed the best on each benchmark data set. Values highlighted in black are where MVMC did as well or better than the best algorithm on that data set. A positive value in the difference in the number of clusters indicates MVMC produced more clusters than the benchmark clusters and vice-versa.

The MVMC algorithm is able to match or slightly exceed some of the best results for the benchmark data sets. While MVMC is not clearly the dominant method across all of the methods, it is often close to the performance of many of the best methods, across all of the data sets. The only data sets where MVMC does not have close performance to the best method on that data set are the BlogCatalog and Olympics data sets. For both of these data sets the MVMC method tends to produce too many clusters relative to the benchmark clusters, while those methods that do have state-of-the-art performance on these data sets require the number of clusters as part of the algorithms. In the case of the BlogCatalog data set the data set has unusually few clusters relative to the number of objects and vice-versa for the Olympics data set. So, part of the difference in performance comes down to the number of benchmark clusters being some disjoint with the number of objects that is to be clustered, which is accounted for by a method that requires the number of clusters and not accounted for by one that does not, like MVMC. So, while the MVMC may not be fully better than the various state-of-the-art methods across all of the data sets, it is able to achieve close to state-of-the-art performance and do so across all of the different data scenarios.

Looking more broadly across the different techniques and data sets, MVMC produces the best results on average. The following table, Table 3.4, summarizes the average performance of all of the techniques across all of the data sets.

Method	Average ARI	STD ARI	Average AMI	STD AMI	Within Top Three of ARI	Within Top Three of AMI	Best of ARI	Best of AMI
GP-MGLA	0.442	0.217	0.523	0.213	0	0	0	0
CSPA+	0.441	0.306	0.508	0.281	0	0	0	0
BGPA+	0.481	0.197	0.550	0.191	0	0	0	0
LWBG	0.448	0.252	0.509	0.228	0	0	0	0
MCLA+	0.377	**0.150**	0.453	**0.165**	0	0	0	0
LWMC	0.427	0.220	0.490	0.205	0	0	0	0
DREC*	0.509	0.305	0.555	0.295	3	3	2	2
LPMMC	0.448	0.291	0.526	0.281	0	0	0	0
DISC_A*	0.502	0.283	0.570	0.263	3	3	0	0
DISC_M*	0.344	0.277	0.448	0.290	0	0	0	0
DISC_R*	0.426	0.223	0.525	0.223	1	0	0	0
DICL_A	0.473	0.247	0.546	0.212	1	1	0	1
DICL_M	0.501	0.273	0.585	0.263	1	1	1	0
DICL_AG	0.503	0.212	0.567	0.203	2	1	1	0
DIMC_A	0.494	0.290	0.555	0.284	2	2	0	0
DIMC_M	0.533	0.260	0.629	0.208	1	0	0	2
MSIM_C	0.304	0.246	0.382	0.223	0	0	0	0
CVIC	0.558	0.209	0.612	0.185	2	2	1	0
CNDC_C1	0.533	0.237	0.593	0.195	3	2	2	2
CNDC_C2	0.540	0.215	0.601	0.202	1	0	0	0
CNDC_I	0.537	0.236	0.583	0.209	3	2	0	1
SFI*	0.295	0.268	0.402	0.282	0	0	0	0
SPSL*	0.416	0.384	0.459	0.412	2	1	0	0
CG*	0.523	0.200	0.592	0.190	1	1	0	0

ResK*	0.089	0.173	0.246	0.183	0	0	0	0
IPMMC	0.603	0.212	0.672	0.182	3	5	1	2
NF-CCE*	0.422	0.181	0.522	0.146	0	1	0	0
ETL-MSC*	0.501	0.214	0.567	0.177	4	5	2	2
MVMC	**0.651**	0.219	**0.694**	0.193	**7**	**8**	**3**	**4**

Table 3.4: Performance summary of the techniques across all of the data sets. The top three techniques for each summary measure are highlighted in green and the top performing techniques is in bold font. The techniques are listed in three different sections. From top to bottom they are late integration, hybrid integration, and intermediate integration. Techniques with an asterisk (*) by them indicate that the number of clusters must be supplied to the algorithm. The performance measures of ARI and AMI were averaged across all of the data sets and the how many times a technique was the best or in the top three for a data set is also recorded. The newly proposed technique of MVMC shows the best performance on average.

From the table, it can be observed that MVMC produces the best average performance across all of the data sets. Additionally, of all the techniques investigated in this empirical study, MVMC is also most often the best or in the top three performers across all of the data sets. So, while it is not always the best performing across every data set, it frequently shows superior performance and has a good performance on average. To get a better sense of the relative performance of the different techniques, the following figure, Figure 3.2, displays a color plot of the relative performance on each technique, for each measure across all of the data sets. Green indicates better performance on that data set and measure and red indicates worse performance and on that data set and measure.

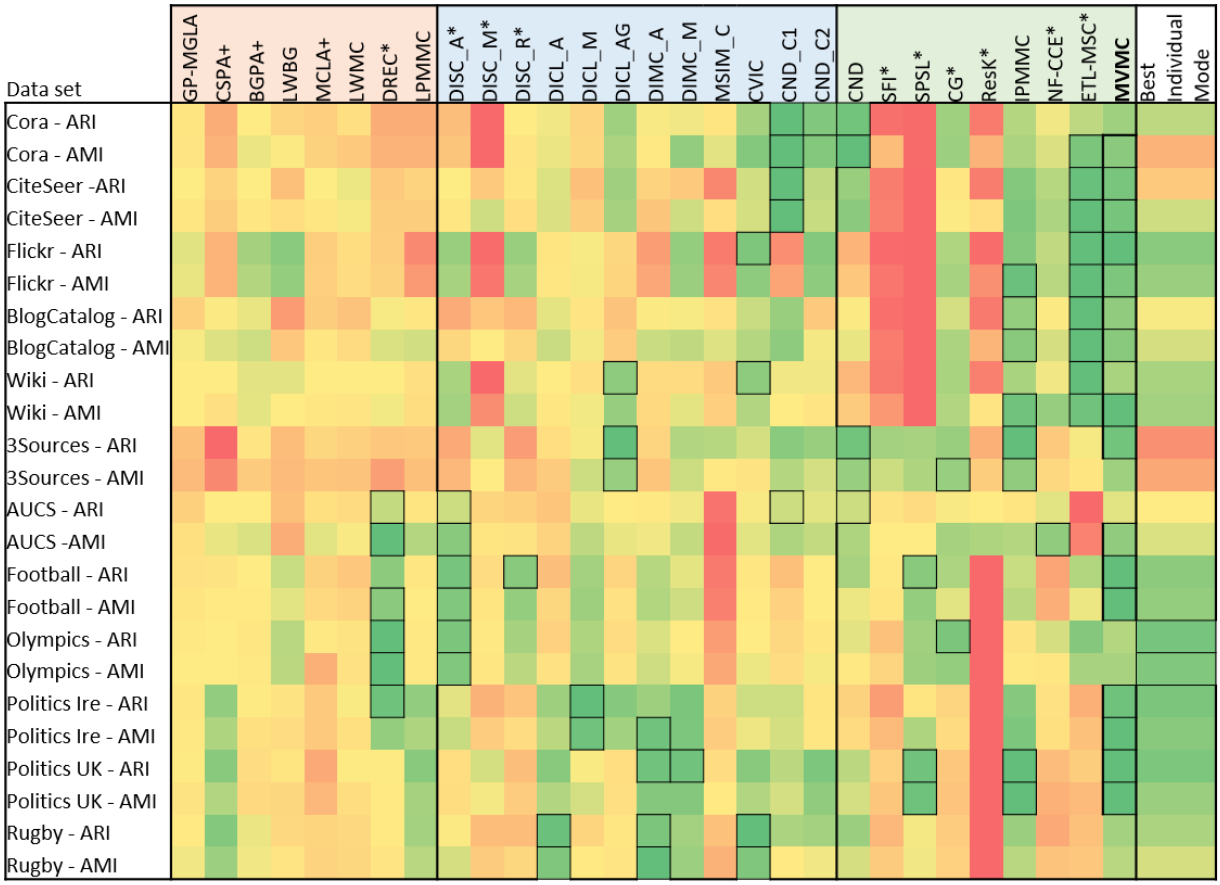


Figure 3.2: Qualitative performance overview of the different multi-view clustering techniques. Green indicates a better performance, relative to the other techniques and red indicates a worse technique. Color squares with a bolded border indicate that technique in the column achieved one of the top three performances on that data set. Across the top, the techniques are broken into late integration (light red shading), hybrid integration (light blue shading), and intermediate integration (light green shading). Techniques with an (*) indicate that that technique requires the number of clusters to be specified. Finally, techniques in bold font are those that had the highest average performance across all of the data sets.

As was indicated by the averages, the MVMC technique often shows good performance across all of the social-based data sets. It is also notable that the MVMC has superior performance over its progenitor algorithm of IPMMC, especially for the publication data sets and some of the Twitter data sets.

MVMC Performance Results

Having evaluated MVMC in terms of the qualities of the clusters produced on various social-based, multi-view data sets, I now turn to the evaluation of the performance of the technique. Since MVMC is a modularity-based, iterative algorithm it would be of interest to understand how many iterations it takes to converge (or not converge) as well as what modularity it can achieve on the various data sets. Additionally, one of the core components of MVMC is its ability to automatically determine the best resolution and weight parameters, which could provide insight

into the data sets in terms of the best views for the multi-view clustering structure. For example, having higher weighted views may indicate that those views are more important to the cluster structure of the multi-view data. The following table, Table 3.5, summarizes the performance characteristics of MVMC across all of the benchmark data sets.

Data set	View	View Resolutions	View Weights	Average number of iterations	Average of iteration when best clustering ocured	Optimal weighted Modualrity
Cora	network	1	1	20	1	1.043
	text	1	1			
CiteSeer	network	1	1	20	1	1.087
	text	1	1			
Flickr	network	1.52	0.62	19.5	2.4	0.59
	attributes	3.11	1.38			
BlogCatalog	network	1.23	0.53	19	5.3	0.77
	attributes	2.52	1.5			
Wiki	network	1.8	1	18.75	0.6	1.37
	text	1.48	0.999			
3Sources	BBC -text	1.5	1	6.1	6.1	1.31
	Reuters -text	1.46	1.005			
	Guardian -text	1.47	1			
AUCS	lunch	1.26	1.21	3	3	2.76
	facebook	1.22	0.47			
	coauthor	0.85	1.73			
	work	1.48	0.75			
	liesure	1.16	0.83			
Football	list	3.48	1.1	4.9	4.9	1.419
	text	1.91	0.44			
	follows	5.21	1.09			
	mentions	5.46	1.14			
	retweets	5.61	1.23			
Olympics	list	4.02	0.99	8	8	1.45
	text	3.25	0.8			
	follows	5.99	1			
	mentions	5.88	1.01			
	retweets	6.82	1.19			
Politics Ire	list	1	1	20	1	2.13
	text	1	1			
	follows	1	1			
	mentions	1	1			
	retweets	1	1			
Politics UK	list	0.508	1.71	2	2	2.39
	text	0.974	0.38			
	follows	1.16	1			
	mentions	1.19	0.784			

	retweets	0.916	1.129			
Rugby	list	0.88	1.14	13	11	1.58
	text	0.95	0.67			
	follows	1.05	1.03			
	mentions	1.03	1.025			
	retweets	1.02	1.12			

Table 3.5: Performance metrics of the MVMC algorithmn across all of the benchmark data sets. The algorithm was allowed a maximum of 20 iterations, so those data sets that have an average of 20 iterations indicate that the algorithm did not converge to a stable resolution and/or weight and instead selected the clustering with the highest modularity as the best clustering (which was often the one at the initial values).

The performance of the MVMC algorithm produces some interesting results across the different data sets. First, the converged resolutions and weights provide insight into the data sets. Generally, the denser view graphs within a data set have higher resolution parameters. This directly mirrors the intuition behind resolution in modularity [50]. However, it is also interesting to note that some of the smaller-sized data sets like Football and Olympics also have some of the highest resolution values across their views relative to other data sets. This result reflects findings by other authors that using quick heuristics to set resolution parameters (like scaling up the resolution parameter for larger, denser graphs) is not the best strategy, and that similarly sized data sets can have very different topologies and thus need very different resolutions [95]. Turning to the weights, there is a general pattern of having view weights that reflect the view's individual clustering quality. For example, on both the Flickr and BlogCatalog data sets, clustering just the graph of the interactions view produces very poor clusters, while clustering just the tags view produced reasonable clusters. And, from Table 3.5, one can observe that the weights for the tags views are close to double the weights for the network views. So, for any given data set, one can use the final view weights to gain insight into the relative importance of the different views to the multi-view clustering structure.

Second, MVMC does not converge to stable resolutions or weights across all of the data sets or all of the runs within a particular data set. In particular, the method does not converge for the Cora, CiteSeer, or Politics Ire data sets. In these cases, the initialized resolutions and weights of one end up producing the highest modularity results, and thus become the final clustering. Also, for some data sets like Flickr, BlogCatalog, and Wiki the method often did not converge until it was close to its maximum number of iterations, resulting in the converged resolutions and weights being used some of the time and the initial resolutions and weights being used other times. The convergence behavior of these data sets mirrors what other authors have observed in terms of having networks with difficult to recover cluster structure [101]. So, from these initial results, the convergence of the MVMC algorithm gives insight into the cluster nature of the data set. To investigate this result further, the following table, Table 3.6, shows the Pearson Correlation Coefficient between the number of iterations and the various cluster goodness metrics.

	Within MVMC Method	Across All Methods
Correlation between ARI and number of iterations	-0.584	-0.648

Correlation between AMI and number of iterations	-0.702	-0.719
Correlation between number of iterations and modualrity	-0.619	N/A

Table 3.6: Correlation between the number of iterations used by MVMC and the clustering performance of both MVMC and the average across all techniques. There is a strong negative correlation indicating that the more iterations it takes MVMC to reach convergence the worse the clusters found are and the worse the clusterability of the data set is.

The number of iterations of MVMC has a moderate to strong negative correlation with clustering performance. That is to say, that the greater the number of iterations MVMC takes to converge (or not converge) the lower quality will be the clusters that are produced. This effect applies not just to the clusters produced by the MVMC method, but to the clusterability of a data set as well. There are moderately strong negative correlations between the number of MVMC iterations and the average of the ARIs and AMIs across all techniques for each data set. So, the convergence of the MVMC method not only gives a reasonable diagnostic on whether the method is producing quality clusters, but also how clusterable the data set is as well.

There is one notable exception to the aforementioned pattern, which is skewing the correlation to be less negative, and that is the Politics Ire data set. For this data set the method alternately iterates between two stable configurations of resolutions and weights. One configuration of resolutions are all less than 1 and the other are all larger than three. So, the technique never hits its convergence criteria, even though it does find stable resolution and weight settings. So MVMC uses the maximum iterations, when it should actually be much less, and does actually get good performance on the data set. Exploring this phenomena further, it was found that changing the initialization of the resolution parameters does not prevent this alternating pattern from happening. So, the Politics Ire data set seems to contain two distinct regions and weights of modularities that are also quite distinct from each other. So, while the number of iterations does not always guarantee its clustering performance or the clusterability of the data set, it does always provide some insight into cluster structures present within the data set.

Taken together, these results suggest that the performance measures of the MVMC algorithm can be used as evidence-based measures for understanding the quality of the clusters produced. First, the weighted combination of the modularities from all of the views gives a measure of how well the MVMC method performed and the quality of the clusters produced, much like standard network modularity. Second, the number of iterations taken to reach convergence (or, not reach convergence) is also a strong indicator of the clustering structure present in the data. The more iterations MVMC takes to reach convergence, the weaker the cluster structure present in the data is. This evidence-based convergence measure is one of the benefits of the iterative algorithm used in the MVMC procedure. Third, and finally, the relative weights of the MVMC procedure illustrate which of the views are most important to the clustering that is produced. Just as the spectral adaptive view weights indicate which of the views are the most important to clustering, so do does the adaptive view weights in the MVMC algorithm. So, one of the distinct

benefits of the MVMC algorithm are that it provides evidence-based indicators of the clustering performance as well as which views are the most important to a clustering, without having any knowledge of any labeling scheme for the data.

MVMC with Different Weight Initialization

In all of the previous tests, the resolutions and weights of the MVMC method were all initialized to one. In this section, I will analyze changing the initialization of these parameters in way that incorporates some knowledge form the view graphs. In particular, it was observed in the previous chapter that the denser view graphs were more often then not the less useful graphs for multi-view clustering. Part of this reason is that the denser graphs were usually the derived view graphs of data like text, instead of the naturally occurring view graphs. So, in these test the view weights are initialized as:

$$w^v = \frac{sparsity(A^v)}{max(sparsity(A))} \tag{3.18}$$

So, the sparsest view weight gets an initialized weight of one, and all the other views get proportionately less weight based on how sparse they are compared to the sparsest view. The following table, Table 3.7, summarizes the difference in results of using this sparsity-based weight initialization from the standard all-one weight initialization.

Data set	View	Difference in final resolutions	Difference in final weights	Difference in ARI	Difference in AMI
Cora	network	-1.69	-0.1	0.1	0.1
	text	-0.71	0.1		
CiteSeer	network	-1.67	-0.18	0.15	0.09
	text	-0.71	0.18		
Flickr	network	0.03	-0.05	0.04	0.06
	attributes	0.09	0.05		
BlogCatalog	network	-1.32	-0.01	0.5	0.42
	attributes	-8.32	0.04		
Wiki	network	-3.37	0.02	0.014	-0.01
	text	-1.72	-0.021		
3Sources	BBC -text	-0.09	-0.002	0.06	0.01
	Reuters -text	-0.09	0.015		
	Guardian -text	-0.1	-0.007		
AUCS	lunch	0	0.01	0.002	-0.001
	facebook	-0.01	0		
	coauthor	-0.01	-0.01		
	work	0	0		
	liesure	-0.01	0		
Football	list	0	0	0	0
	text	0.01	0		
	follows	0	-0.01		
	mentions	-0.01	0		

	retweets	0	0		
Olympics	list	0	0	-0.01	-0.01
	text	0	0		
	follows	0.09	-0.005		
	mentions	-0.02	-0.002		
	retweets	-0.01	0		
Politics Ire	list	-0.02	0	0.174	0.07
	text	-0.13	0.497		
	follows	-0.57	-0.102		
	mentions	-0.68	-0.07		
	retweets	-0.56	-0.41		
Politics UK	list	0.008	0	0	0
	text	0.004	0		
	follows	-0.01	0		
	mentions	0	0.004		
	retweets	-0.004	0.009		
Rugby	list	-3.28	0.06	0.09	0.09
	text	-1.86	-0.02		
	follows	-5.37	0		
	mentions	-5.44	-0.015		
	retweets	-6.38	-0.04		

Table 3.7: Comparison of results of the MVMC algorithm with different weight initializations. A negative value indicates that the sparsity-based initialization produced higher values and vice-versa for the positive values. So, positive values Difference in ARI and AMI columns indicate the the standard initialization MVMC prodcued better results and vice-versa.

The first result to note is that there are differences in both clustering qualities and algorithmic metrics between the two different initializations. this would suggest that the initialization does matter in terms of the final outcome. Starting from a different weight or resolution could cause the algorithm to converge to different final weights and final resolutions. This idea of having different stable regions of resolutions and weights has also been observed by other authors [128], [138]. Also, some of the data sets with stronger cluster structures (i.e. they have generally good performance across a range of clustering techniques and paradigms) like Politics UK or Football tend to have much less difference between the initializations of MVMC in both the ARI/AMI performance measures and the MVMC performance measures of view resolutions and view weights. This would suggest that the results here conform to the results from other authors where they have found that different data sets have very different stable regions of resolutions and that the greater (and fewer) stable regions that exist, the more clusterable the data set is [138]. The second result to note is that the standard initialization of having the weights and resolutions all equal to one produces consistently better results in terms of cluster quality.

3.4.2 Multi-view Density-Modularity Clustering Results

In this section, the ability of a density-based modularity (Constant Potts Model) to cluster multi-view social-based data from a limited MVMC algorithm variant, Multi-view Density-Modularity

Clustering (MVDMC) is empirically investigated. As with the MVMC algorithm, the MVDMC algorithm was run twenty times across all of the data sets and the average results across all of the runs and the data sets were collected. The following table, Table 3.8, displays MVDMC's peformance on the benchmark data sets relative to the baseline modularity method of IPMMC.

Data set	ARI	AMI	Difference with IPMMC ARI	Difference with IPMMC AMI
Cora	0.3	0.39	-0.03	-0.01
CiteSeer	0.38	0.36	0.01	-0.01
Flickr	0.038	0.28	-0.532	-0.36
BlogCatalog	0.32	0.39	-0.31	-0.23
Wiki	0.32	0.46	0.03	-0.03
3Sources	0.73	0.75	-0.03	0.01
AUCS	0.57	0.61	0	-0.08
Football	0.75	0.87	0.1	0.04
Olympics	0.67	0.82	0.13	0.05
Politics Ire	0.81	0.77	-0.08	-0.09
Politics UK	0.92	0.88	-0.07	-0.09
Rugby	0.64	0.69	-0.01	0.01

Table 3.8: Performance of MVDMC and the baseline modularity maximization algorithm IPMMC on their ability to produce good clusters relative to the benchmark clusters. MVDMC often has performance close to IPMMC indicating that density-based modularities can also be used in the proposed methodology.

Generally, the MVDMC algorithm has very similar performance across the benchmark data sets. The two notable exceptions are Flickr and BlogCatalog where MVDMC has significantly worse performance. So, it would seem that using a density-based modularity is also capable of producing reasonably good results across many social-based data scenarios within the proposed framework of this chapter. There are, however, two data sets that are notable exceptions to this performance. To investigate these results further and better understand the performance of using a density-based modularity, the following table, Table 3.9, records the results of MVDMC algorithm's performance against the best method for each data set and MVDMC's number of clusters.

Data set	ARI	Difference with Best ARI	AMI	Difference with Best AMI	Avg. number of clusters	Difference in number of clusters	Best Method
Cora	0.3	-0.14	0.39	-0.1	25.2	18.2	CNDC_C1
CiteSeer	0.38	-0.05	0.36	-0.05	42.6	36.6	CNDC_C1
Flickr	0.038	-0.592	0.28	-0.37	239.5	230.5	ETL-MSC
BlogCatalog	0.32	-0.42	0.39	-0.32	61.9	55.9	ETL-MSC
Wiki	0.32	-0.02	0.46	-0.03	61.1	44.1	ETL-MSC
3Sources	0.73	-0.03	**0.75**	**0.01**	3.7	-2.3	IPMMC
AUCS	0.57	-0.1	0.61	-0.19	6.3	-2.7	DREC
Football	0.75	-0.08	0.87	-0.02	18	-2	DISC_A
Olympics	0.67	-0.23	0.82	-0.12	21.7	-6.3	DREC
Politics Ire	0.81	-0.11	0.77	-0.1	13.3	6.3	DICL_M

Politics UK	0.92	-0.07	0.88	-0.09	11.7	6.7	IPMMC
Rugby	0.64	-0.08	0.69	-0.01	25.5	10.5	CMIC

Table 3.9: Summary of the performance comparison to the clusters produced by MVDMC and those produced by the method that performed the best on each benchmark data set. Values highlighted in black are where MVDMC did as well or better than the best algorithm on that data set. A positive value in the difference in the number of clusters indicates MVDMC produced more clusters than the benchmark clusters and vice-versa.

As would be expected from MVDMC's performance relative to IPMMC, MVDMC's performance against the best methods for each of the data sets mirrors the performance of IPMMC against the best methods for each of the data sets. The two notable exceptions of much worse performance are Flickr and BlogCatalog. Turning to the number of clusters found by MVDMC for these data sets, it is clear that MVDMC heavily over-clusters the data. In fact, more often then not, MVDMC tends to produce more clusters than the benchmark labels would indicate for each of the data sets. So, while it seems using a density-based modularity can produce good results, it also has the tendency to over-cluster.

Next, I analyze the performance of the MVDMC algorithm internally. To get a better understanding of the performance of the MVDMC algorithm, the following table, Table 3.10 displays the various internal metrics of performance for MVDMC across all of the benchmark data sets.

Data set	View	Mode of view resolutions	Number of iterations used	Optimal weighted modularity
Cora	network	0.0001	25	1.01
	text	0.0001		
CiteSeer	network	0.0001	25	1.07
	text	0.001		
Flickr	network	0.05	25	0.29
	attributes	0.0001		
BlogCatalog	network	0.01	25	0.61
	attributes	0.01		
Wiki	network	0.0001	25	1.4
	text	0.01		
3Sources	BBC -text	0.0001	125	1.6
	Reuters -text	0.05		
	Guardian -text	0.05		
AUCS	lunch	0.0001	3125	2.4
	facebook	0.001		
	coauthor	0.05		
	work	0.05		
	liesure	0.05		
Football	list	0.05	3125	2.3
	text	0.05		
	follows	0.05		
	mentions	0.05		
	retweets	0.05		

Olympics	list	0.01	3125	2.738
	text	0.05		
	follows	0.05		
	mentions	0.01		
	retweets	0.05		
Politics Ire	list	0.05	3125	2.08
	text	0.001		
	follows	0.05		
	mentions	0.001		
	retweets	0.05		
Politics UK	list	0.0001	3125	1.89
	text	0.05		
	follows	0.001		
	mentions	0.05		
	retweets	0.005		
Rugby	list	0.001	3125	2.8
	text	0.01		
	follows	0.05		
	mentions	0.05		
	retweets	0.05		

Table 3.10: Performance metrics of the MVDMC algorithm across all of the benchmark data sets. Since there is no means of determining the resolutions and/or weights automatically for Constant Potts Model modularity, the number of iterations is the combination of number of views and possible resolutions. The modularity is the standard network modularity and not the Constant Potts Model modularity

From the MVDMC performance metrics a couple of interesting results emerge. First, as was suspected and pointed out in the methods section, different views of the same data often require different resolutions, even with density-based resolutions. So, when designing an density modularity-based algorithm for clustering multi-view, social-based data, it is necessary to allow the resolution parameter to vary between views. Also, the modularity obtained by the MVDMC procedure also correlates strongly with the performance of the method relative to the benchmark clusters. It should be mentioned again that this modularity is the standard network modularity and not the Constant Potts Model that is used to cluster the graphs. So, in the absence of labels it is possible to get some indication of the clustering quality found by using a density-based modularity by looking at the standard network modularity.

3.4.3 Comparison Between Modularities

To conclude the empirical testing section I analyzed the differences between the MVMC algorithm and the experimental MVDMC algorithm. The following table, Table 3.11, summarizes the difference in goodness measures and number of clusters found between the two methods across all of the benchmark data sets.

Data set	ARI Difference between MVMC and MVDMC	AMI Difference between MVMC and MVDMC	Number of Clusters Difference between MVMC and MVDMC
Cora	0.06	0.05	-9.75
CiteSeer	0.01	0.02	-10.9
Flickr	0.592	0.33	-31.75
BlogCatalog	0.32	0.23	-102
Wiki	-0.026	0.038	-18.5
3Sources	0	-0.03	-1.35
AUCS	0.062	0.129	0
Football	0.12	0.05	-0.1
Olympics	0.06	0.05	0
Politics Ire	0.104	0.11	-3.9
Politics UK	0.07	0.09	0
Rugby	0	-0.01	-9.5

Table 3.11: Comparison between the performance of the MVMC and experimental MVDMC algorithms. Negative values indicate that the MVDMC algorithm had a greater value for the measure than MVMC and vice-versa for positive measures. Generally, MVMC has better performance in terms of cluster goodness and does not over-cluster the data like MVDMC often does.

Generally, the MVMC method with its use of Reichardt and Bornholdt modularity performs better than MVDMC with its use of a density-based modularity. While in many ways this is an unfair comparison, since the former has equations to precisely set the resolution and weight parameters, and the later does not, it is also easy to see that the Constant Potts Modularity has a tendency to over-cluster data. So, while using a density-modularity to cluster multi-view, social-based data can often produce reasonable results, it can also easily over-cluster the data. As a final comparison between the two methods the following table, Table 3.12 presents the performance summaries for the two methods.

Comparison measure	MVMC	MVDC
Average ARI	**0.652**	0.537
Average AMI	**0.694**	0.606
Avg. of absolute difference between number of found clusters and number of benchmark clusters	**1.257**	9.816
Correlation between modularity and ARI	0.464	**0.769**
Correlation between modularity and AMI	0.575	**0.816**

Table 3.12: Performance summaries for MVMC and MVDMC. Bolded values indicate which method had better performance for the given metric.

From the results, MVMC has consistently better performance. Also, both methods produce modularities that correlate reasonably well with actual performance across all of the benchmark data sets. This suggests once again that the modularity found by a modularity-maximization method can be a good evidence-based measure to judge the quality of labels found in the absence of having any labels. Also, MVDMC has higher correlation with performance and modularity than does MVMC. Since MVDMC is using standard network modularity in its cluster evaluation, this result suggests that another evidence-based measure for determining cluster goodness in the absence of labels with modularity-maximization techniques is to use the standard network modularity, without any resolution or weight parameters.

3.5 Discussion

In this chapter I investigated the use of modularity as an objective function for characterizing clusters in multi-view data. The empirical testing across many different benchmark, social-based, multi-view data sets demonstrated that the adapted modularity function does indeed perform well as an objective function for finding clusters within the data. In particular, having a good modularity value for a multi-view clustering correlates with finding good clusters within the data. In addition, it was observed in the empirical testing of modularity that the nature of the view graphs can greatly effect the cluster quality with everything else being kept the same. In this work, it has generally been observed that creating k-Nearest Neighbor graphs directly from the view features tends to give the best results (as opposed to creating cluster association graphs from the view clusters), but occasionally incorporation of other elements of information can improve the cluster structure of these graphs. So, how to best create graphs from the different views to best find clusters within the data remains an open area for future research.

Since it is well known in the Network Science community that the use of modularity for clustering can be further improved by setting appropriate resolutions for the modularity function, I next embarked on crafting an algorithm that could take into account resolutions and weights from different views. Using previous works demonstrating the means to set resolution and weight parameters given clusters and a planted partition model of the clusters in the networks, I have proposed a new method of clustering multi-view, social-based data that can use any types of data in the views. The overall procedure consists of transforming all of the data into a graph space, which is common in multi-view clustering of image data, and then using an iterative modularity maximization procedure the optimizes a view-weighted, resolution corrected version of the network modularity function. In order to perform the optimization, I used an alternating, iterative scheme that alternated between updating clusters assignments, and then updating the view weights and resolutions. This optimization procedure leverage a means of approximate fast graph clustering as well as analytic functions. It would be interesting for future research to see if there are other means of optimizing the the special network modularity function proposed in this chapter. Using an iterative, alternating procedure that alternates between clustering the view graphs and updating the resolutions and weights for each of the views showed improved empirical performance over just using modularity without setting these parameters. Additionally, the proposed algorithm was also close to, or as good as state-of-the art across all the social-based data scenarios tested in this work. Thus, the proposed technique can not only get state-

of-the-art performance in some social-based data scenarios, but always gets good performance for any social-based data scenario. No other technique investigated in this work could achieve both results. Finally, the proposed technique is relatively computationally fast. The two most computationally demanding aspects of the method are the construction of the view graphs and the clustering of the view graphs. Both of these steps have fast, approximate techniques that can also scale to very large (i.e. millions of objects) data sets. And, the construction of the view graphs is only done once and the clustering of the view graphs is usually only done a few times for a reasonably clusterable data set, which makes the computational demand low in practice.

Another means of correcting for the resolution in modularity is to use a 'resolution-free', density-based modularity like Constant Potts Model modularity. In investigating the use of these modularities for the practical clustering of multi-view, social-based data, some interesting results observed. First, there is not automatic means of setting the resolution parameter for these modularities, since the resolution parameter is related to cluster densities which typically monotonically increase the number of clusters increases. It does, however, seem possible to design a function that could balance between obtaining high cluster density against the number of clusters, and this is left to future work. It was also observed in testing Constant Potts Model modularity that it tends to over-cluster data sets which could be another issue with the function to adress in future work.

In addition to the good performance of the MVMC technique, the technique also provided some beneficial byproducts from a practical use standpoint. For one, the view weights found by the procedure provide a reasonably reliable evidence-based measure to understand which views were more important to the cluster structure. Second, the number of iterations that MVMC takes to reach convergence (or not reach convergence) provided a reliable evidence-based measure for the quality of the clusters produced as well as indicated how clusterable the data set was. This result suggests that data sets which have more stable and resolutions and weights also tend to have a stronger cluster structure. This result confirms what other works have found both empirically and theoretically concerning resolution in graphs. If a community structure exists across a wide range of possible resolution values then it is a strong community structure. Third, much like network modularity itself, the weighted combination of the view modularities also gives a good, evidence-based indication of the strength of the cluster structure. Taken together, all of this suggests one of the distinct benefits of MVMC is in its ability to explore patterns in structure within data. Not only does it produce generally good cluster structures, but it also gives evidence-base measures about the quality of the cluster structures as well as the importance of different views relative to the multi-view cluster structure.

Chapter 4

Case Study: Characterizing Communities of Hashtag Usage on Twitter During the 2020 COVID-19 Pandemic

In this chapter I use a previously developed multi-view clustering technique in order to cluster hashtags from twitter data collected on the COVID-19 pandemic. Hashtags are a social media innovation that allows users to find and participate in discussions of interest. So, clusters of hashtags can often give unique insight into topical focal points for social media users. In this chapter, I perform the first ever multi-view clustering of hashtags from twitter data collected during and about the COVID-19 pandemic. The results of the multi-view clustering demonstrate that there are temporal patterns and changes in how users use Hashtags over the course of a pandemic. The multi-view clusters also give distinct insight into the topical focus areas for twitter users over the course of the pandemic and how different topics have different user bases.

4.1 Background

At the time of the writing of this thesis the world is undergoing a pandemic. This pandemic, which is caused by the SARS-CoV-2 virus and often referred to as the COVID-19 pandemic, has caused immense societal and economic disruption across the world. Since the onset of the COVID-19 pandemic many nations have adopted a social-distancing strategy which has had the unintended consequence of emphasizing and increasing the role of social media in linking people together[8], [72]. Consequently, the study of social media data during the current COVID-19 pandemic can provide unique insights into online social behavior.

The first, and perhaps the most important, study into online social behavior during the COVID-19 pandemic is studying how information and misinformation operates during a pandemic. Good information is a key enabler to combat the effects of the pandemic whereas misinformation can exacerbate its effects [72], [52]. Recent studies into the prevalence and persistence of misinformation have shown that misinformation on the COVID-19 pandemic has been especially persistent and spreads through online social networks quickly [145], [8], [17]. The spread of COVID-19 misinformation has become so problematic and widespread that many many researchers are

referring to it as an 'Infodemic' [52], [33], [8]. The Infodemic is characterized by a virus-like spread of misinformation across many different communication mediums, most notably online social networks.

In the same line of research, other researchers have identified important mechanisms by which the misinformation propagates in social media. Recent research has identified the importance of bots in the spread of misinformation [48]. Other research has highlighted the role of alternative news sources and user characteristics like political beliefs in the spread of COVID-19 misinformation [17], [63]. Using some of the aforementioned means by which COVID-19 misinformation spreads, one recent article designed an Infodemic susceptibility score and investigates the correlations between countries that are susceptible to Infodemics and the impacts of the actual pandemic on that country [52]. The important mechanisms underlying the Infodemic susceptibility score are both the content that users are exposed and the interactions between users.

One area that is less clear is how social media users may be changing their behavior and how social media communities and discussions are changing in response to the COVID-19 pandemic and Infodemic. While the aforementioned research has shown that social media users are spreading COVID-19 misinformation, sometimes even faster than good information, its not clear how users' interactions or how discussion communities may be changing during the pandemic. It is also not clear if there are any topical areas of focus for social media users during the ongoing pandemic. Are social media discussions focusing around topics like health and welfare or around politics and business, or any combination thereof?

One of the recent social media innovations that have been used to track and understand conversations and conversational topics are hashtags. Hashtags originated in 2007 on the social media platform Twitter as a means of allowing users to efficiently retrieve information relevant to a topic [154]. The use of hashtags on Twitter has expanded to not only be a means of characterizing discussion topics, but also a means of predicting user links and characterizing both communities of users as well as the users themselves [154], [142], [114], [112], [115]. As such, clustering of hashtags can be used to understand topics of interest for social media users and the communities that form around certain discussions [78], [131]. The clustering of hashtags has been done by either the text context used with the hashtags, co-occurrence of hashtags within the same social media post (i.e. same tweet), or by having similar users that use the same hashtags [78], [142]. More recent work on clustering hashtags has focused on better feature engineering of the text that accompanies hashtags in order to capture the semantic meaning of the text and thus better hashtag clusters [131]. To date, no work has attempted to combine all of these different views of hashtag usage and use multi-view clustering to cluster hashtags in order analyze social media data for understanding topics of interest for social media users. Part of the difficulty in multi-view clustering of social media data like hashtags are that the data is often very large (on the order of tens or hundreds of thousands of hashtags being used in any given conversation) which can pose problems for many of the existing multi-view clustering techniques, especially intermediate integration techniques. Additionally, social media data often have partially-complete views of data; social-media users and objects, like hashtags, may not have any interactions within some views. For example, a hashtag may never co-occur with another hashtag, or a user may never engage in an activity like re-tweeting. These partially complete views pose challenges for many existing multi-view clustering techniques as these objects naturally become isolates or small connected components in view graphs. Overall, clustering of hashtags can provide valuable insight into

important topics in a discussion and insight into social media users, and that hashtags have never been clustered by multi-view clustering.

So, in this chapter I use multi-view clustering in order to cluster hashtags from COVID-19 twitter data in order to better understand the discussion topics present during the pandemic and if these topics change over the course of the pandemic. The main contributions of this chapter are summarized as follows:

- The first use of a multi-view clustering technique and approach to understand topical groups of hashtags.
- The first use of a multi-view clustering technique on a large, social-based data set; multiple collections each consisting of upwards of 85,000 objects are clustered in this study whereas most multi-view clustering techniques have been used on data sets, social-based or otherwise, of at most one collection of 70,000 objects (i.e. full MNIST data set).
- Discovery of a temporal trend in hashtag usage that displays three distinct periods of different hashtag usage and topical clusters over the course of the COVID-19 pandemic, from the 1st of February 2020 to the 30th of April 2020.
- Characterization of topical clusters of hashtags that give distinct insight into what conversations surround the COVID-19 pandemic on twitter, such as co-opting of the calamity to support different causes and a persistent coupling of U.S. politics related hashtags with conspiracy theory related hashtags.

4.2 Data and Methodology

In this section, I will detail the data used in this investigation of the use of COVID-19 hashtags and the methodological set up to find clusters of hashtags. To cluster the hashtags, I will employ a multi-view clustering technique — specifically, Multi-view Modularity Clustering (MVMC) — in order to exploit the richness of the data in providing for multiple possible views of understanding clusters of hashtags.

4.2.1 Data

The data for this analysis comes from Twitter's streaming API [1]. The data collection was done using a list of key words including "coronavirus", "coronaravirus", "wuhan virus", "wuhanvirus", "2019nCoV", "NCoV", "NCoV2019" [63]. The collected data spans the time period from 1 February 2020 to 30 April 2020 and consists of over 300 million tweets that have, on average, 45,000 unique hashtags per day. The following figure, Figure 4.1 depicts the daily statistics concerning the use of hashtags within the data set. It should be noted that as a prepossessing step, any hashtag that was not used in at least 3 tweets was not included in the data. These hashtags are often misspellings of more widely used hashtags.

[1] https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/basic-stream-parameters

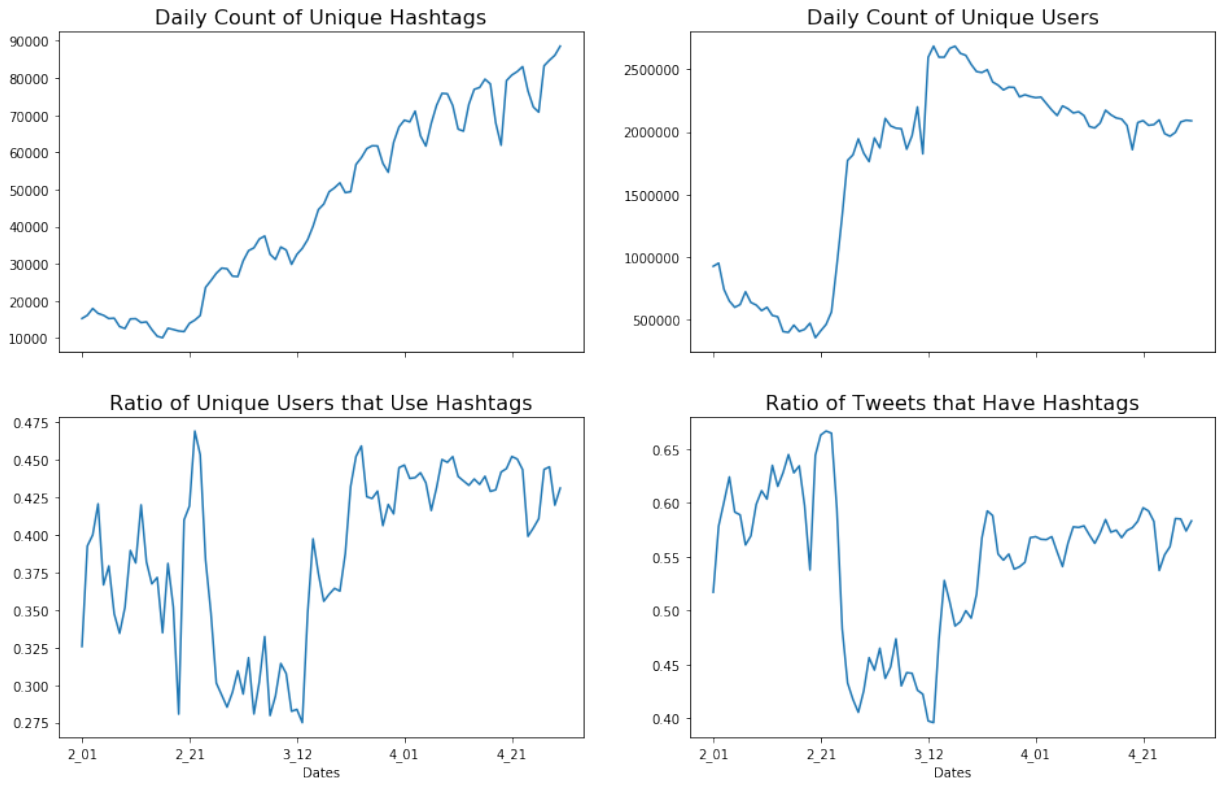


Figure 4.1: Daily Statistics of the COVID-19 Twitter Data from 1 February 2020 to 30 April 2020. Use of Hashtags by users and within tweets remains high and persistent over the time period.

Hashtag usage within the data is both prevalent and increases in the diversity of hashtags being used over time. The use of unique hashtags generally increase over the time period of data collection and displays some weekly cyclical patterns as well (i.e. slight drops in the number of unique hashtags being used on weekend days). It is interesting to note that these counts are counts of unique hashtags and not the total use of hashtags. So, it is possible that as the scope of the COVID-19 pandemic expanded, hashtags that were originally unassociated with the COVID-19 pandemic end up becoming a part of the conversation. It is also possible that as the scope of the pandemic expanded, that new hashtags were invented to better address the changing needs of the conversation about the pandemic. Additionally, there is a relatively high ratio of hashtags being used in tweets throughout all of the data (greater than 40% of tweets have a hashtag). While this is in part due to the means of collecting the data, it also reflects a trend observed by other authors of increasing hashtag usage among social media users generally [115], [154]. The ratio of hashtag usage in tweets has three observable phases over time. In the first phase, from 1 February to 24 February, the ratio of hashtags present in tweets remains at its highest, with a slight positive trend. Then, starting on the 25th of February the ratio of hashtags in tweets decreases and remains lower until the 15th of March. Finally, in the third phase from the 16th of March until the 30th of April, the ratio of hashtags being used in tweets remains stable at around 55% of the tweets. So, while there is a high percentage of tweets that have at least one hashtag, this percentage is not stable over time. While its not clear why this temporal trend exists, it is likely related to the dynamic nature of the users interacting on Twitter over the various stages of

the COIVD-19 pandemic.

There is also a reasonably high percentage of hashtag use among individual users, which generally increases with time. As with the ratio of tweets that feature a hashtag, the ratio of users that use at least one hashtag on a given day breaks into three observable periods. The first period is a period of a lot of oscillation in daily hashtag usage centered around 37%. The second period is a decline in hashtag use corresponding to the same period of decreased hashtags being used in individual tweets, from 25 February to 15 March. The third period is from the 16th of March to the 30th of April and has a sustained hashtag use at around 43% of users using at least one hashtag in a tweet per day. Overall, 50.47% of users use at least one hashtag during the three months that data was collected. In terms of individual hashtag usage, for users in general the usage statistics of hashtags are as follows: min: 0, max: 100%, mean: 31.6%, and standard deviation: 38.8% of their tweets featuring hashtags. Of those users that use at least on hashtag in their tweets, the usage statistics become: min: 1.9%, max: 100%, mean 62.6%, and standard deviation: 32.4% of their tweets featuring hashtags.

This hashtag usage takes place in a background of variable trends in the number of unique daily users present within the data set. There is a declining number of unique users within the data from the 1st of February to around the 21st of February, at which point there is a large increase in the number of unique users until the 20th of March. From the 20th of March to the end of April, the number of daily unique users begins to decline again. So, while the use of hashtags increases as well as the use of unique hashtags increases, the number of unique users actually declines. So, during the early stages of the pandemic there are a fairly small number of users tweeting relevant tweets which often have hashtags which then transitions over the course of the pandemic to a much larger user base that does not initially use many hashtags. This observation suggests that user's interactions on Twitter are dynamic over the course of a pandemic and that discussion topics, in the form of hashtags are also dynamic over the course of a pandemic. It also suggests that hashtag usage becomes more widely adopted after an initial surge in users possibly as means of better characterizing the new and burgeoning conversations happening on Twitter surrounding the pandemic. All together, the nature of the use of hashtags and the hashtags in use have likely changed over the course of the COVID-19 pandemic.

Data Processing

The aforementioned collection of raw tweets was then transformed into view data for clustering of the hashtags. First, the data was separated into days. Second, the daily tweet data was pre-processed as follows: First, any hashtag which was used in less than 3 tweets was excluded from the data. Second, the tweet text was preprocessed by removing all of hashtags, URLs, symbols (i.e. emojis, punctuation, etc.), and twitter-specific tags (i.e. mentions, quotes, etc.) from the tweet text. From there, for each hashtag, the accompanying text, other hashtags, the user, and any URLs used in each tweet that contained that hashtag were extracted. This data was then separated into four separate views of the data.:

- The first view is the text view and contains all of the pre-processed text from each tweet that features a hashtag. The intuition behind this view is that the text accompanying a particular hashtag may give insight into how and what a particular hashtag is used for in

discussions. The accompanying tweet text, whether used in a raw form or given semantic enhancement has been previously used to cluster hashtags [78], [131].

- The second view is the users which tweet a hashtag with the idea that users may be partial to tweeting particular hashtags as part of a discussion. Shared users have also been used to cluster hashtags in previous works [78].
- The third view is the URLs which co-occur with a hashtag. Since URLs are often used as information to support claims in tweets, this view should give insight into what information is underlying the use of certain hashtags.
- The fourth view is the co-occurrence of hashtags within tweets. Within any given tweet using hashtags, a user may use multiple, related hashtags as part of their tweet. The co-occurrence of hashtags is frequently used to create hashtag-to-hashtag networks for analysis by standard network science techniques [131].

So, all together, the collected tweets are processed to create four different views of the data for each day in order to cluster the hashtags.

4.2.2 Methodology

In order to find clusters of the hashtags from the multi-view data, the Multi-View Modularity Clustering (MVMC) procedure detailed in Chapter Three will be used. There are two primary reasons for the use of this technique on this data. First, this technique, which uses a modularity maximization based clustering as its principal means of extracting clusters, is scalable to large data. The collected COVID-19 data often features tens of thousands of unique hashtags being used each day. So, any multi-view clustering technique on the data must be able to handle data of this size. Second, the previous chapter has empirically demonstrated that MVMC is the most generalizable in terms of producing quality results for social-based, multi-view data. And, since there is no accepted clustering procedure for multi-view twitter data scenarios, a technique which is generalizable to many different data scenarios would be desired.

For the MVMC procedure there are two main steps. The first step is to create graphs of each of the views, and the second is to cluster those graphs. For each view, a similarity graph was created. So, for each view graph, an edge represents how similar two objects are with respect to that view. For example, for the text view, an edge indicates that two hashtags share similar text in their tweets, or for the co-occurence view that two hashtags tend to occur with the same set of other hashtags. In order to measure similarity, I first transformed the view data from raw counts (i.e. the number of times a users uses a hashtag) to Term Frequency-Inverse Document Frequency (tf-idf) scores using,

$$w_{ij} = tf_{ij} \times \frac{n}{df_j} \tag{4.1}$$

where tf_{ij} is the number of terms (i.e. users, word tokens, URLs, etc.) occur with hashtag i, and df_j is the number of Hashtags that also feature the jth term. This transformation was done in order to down-weight those terms which are common across all hashtags, like 'COVID19' and up-weight those terms which may be significant for cluster structure. While it has been noted in

previous works that tf-idf can be insufficient for textual information for tweets, the primary reason for this insufficiency that individual tweets have very little text which can accompany them [131], [142]. However, in this methodological set up, many tweets (often hundreds or thousands) are combined for each hashtag, so the text size limitation is not a significant issue. Furthermore, using tf-idf is also very scalable and does not require any kind of *a priori* semantic knowledge database [131]. So, for these reason, I have opted to use tf-idf as a means of processing the view data prior to learning the view graphs. Having applied tf-idf to all of the views, the similarity for each of the views was measured by cosine similarity:

$$s_{ij} = \frac{A_i . A_j}{||A_i|| \times ||A_j||} \tag{4.2}$$

Since the data is large in the number of objects that need to be clustered (i.e. hashtags) — on the order of tens of thousands — the graph learning procedure needs to be a computationally efficient one [106]. For that reason, I have adopted the simple procedure of creating a symmetric k-Nearest Neighbor Graph (k-NN) with the number of nearest neighbors as $k = \sqrt{n}$, where n is the number of objects being clustered [89], [88]. To symmetrize the k-NN the following step as used: once each object has been connected to k of its nearest neighbors, I have adopted the average strategy, $A' = \frac{1}{2}A + A^T$, which is common in spectral clustering methods [106], [150], [158]. So, at the end of the graph learning step, there is a cosine-similarity weighted, undirected graph for each view of the data.

Having learned a graph for each view of the data for each day, the MVMC procedure is then performed for each day. The initial weights and resolutions were all set to one. The convergence tolerance for the resolutions was set to 0.3 and for the weights to 0.1. And, the procedure was allowed to run for a maximum of 20 iterations.

4.3 Results

In this section, I describe the multi-view clustering results on the COVID-19 data. In the first section, I provide an overview of the clustering results. In the second section, I detail the results of learning graphs on the different views of the data and the insights the graphs give about the data and the cluster structures present within the data. In the third section, I analyze the performance of the multi-view clustering technique of MVMC. In the fourth section, I analyze the temporal patterns within the hashtag cluster data. In the fifth section I analyze the user bases of the different clusters of hashtags. Finally, in the sixth section I do an in depth analysis of some of the interesting clusters identified within the data.

4.3.1 Overview of Multi-view Clustering Results

Multi-view clustering of the COVID-19 twitter data extracted between 20 and 160 clusters of hashtags per day, with a varying size of the clusters. These clusters varied in size and also in number between different days. The following figure, Figure 4.2, displays plots of the daily number of clusters and daily cluster size statistics over the full 90 day period. The actual numbers for each day are displayed in the Appendices (Appendix B).

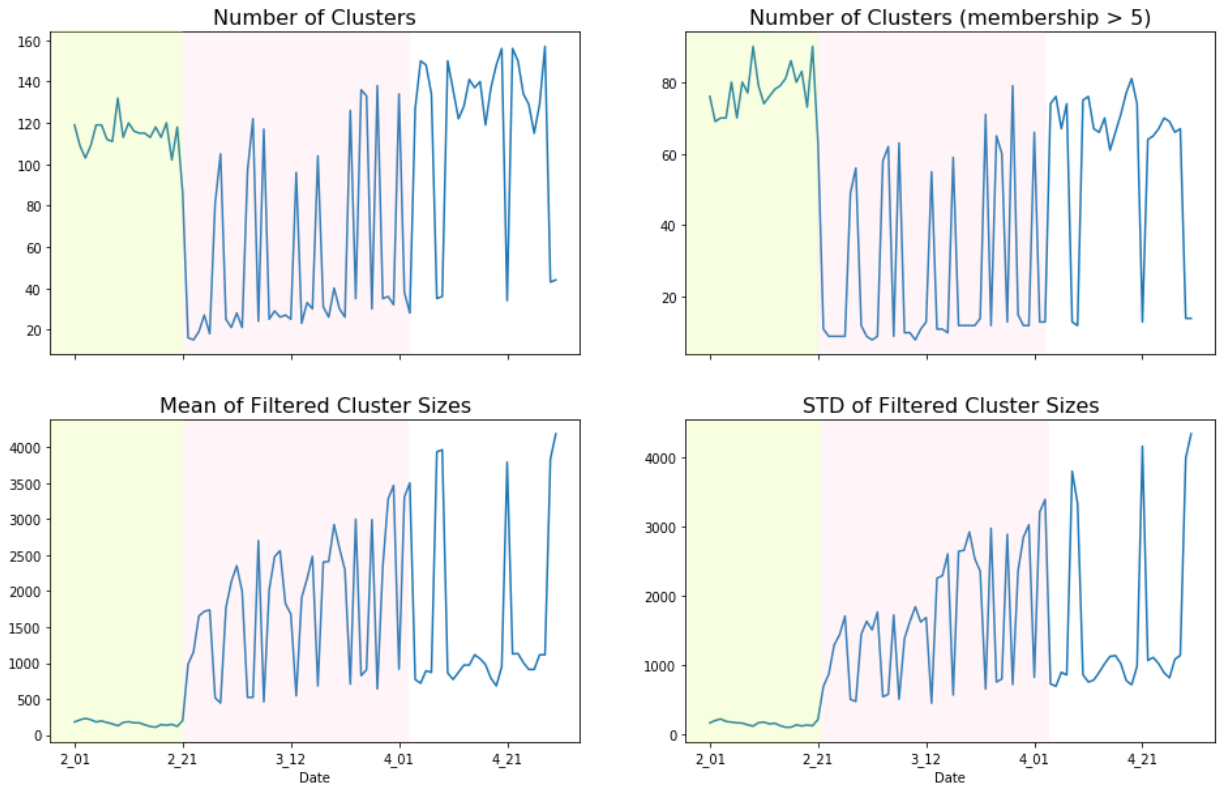


Figure 4.2: Daily clustering statistics on clusters produced by the MVMC technique on four views of the hashtags. The daily clusters display three different, temporal patterns of clustering. The shading in the figures covers those areas which display the three different patterns

There is a presence of several small clusters within the daily clusterings. First, on any given day there were between 20 and 80 clusters that had a size of less than five objects. These clusters were almost exclusively composed of either small, locality-specific hashtags, non-English language hashtags, or hashtags that seem to have little relevance to the pandemic. For example, a particular local business, like a car dealership may have a hashtag that is used in a tweet that happens to mention one of the key terms, which was then retweeted by local residents, which would cause it to be above the initial screening criteria of being used in more than three tweets, but otherwise bear little relation to how any of the other hashtags in the data are used. These small clusters are also often composed of objects that are isolates within one or more of the view graphs, which is what makes them difficult to group into larger clusters. This result illustrates an important point about multi-view clustering of real-world data; the data is often messy and incomplete and requires some degree of additional processing. These arbitrary clusters do not actually contribute much to understanding the data, or the macro-cluster structure of the data, or use of hashtags beyond recognizing that the collection process can produce some noise in clustering results. Removing these small clusters does not affect the overall patterns existing within the clusters, and makes interpretation of the clusters easier.

Second, there are dynamic patterns within the clusters. The clusters start off as many in number and small in size and become fewer in number and larger in size as time passes. Up

until the end of February, there are around 80 non-arbitrary clusters with a size of around 250 hashtags. This pattern changes at the end of February where the number of daily hashtag clusters decreases but the size of these clusters increases. This change in the clustering structure over time may indicate that the use of hashtags and their associated discussions begin to congeal into larger discussions over the course of the pandemic. Additionally, there can be large oscillations in numbers of clusters and sizes of clusters between any given set of days. While, there is an increasing trend toward fewer and larger clusters, there are oscillations present within the data, especially during the middle of the time period, around the month of March. While it is not quite clear why these oscillations occur, it was noted in the description of the data that there are weekly periodic patterns within the number of unique hashtags used over time. So, its possible these oscillations are in part due to cyclical, time-dependent patterns in twitter use. This dynamic nature in the clusterings will be further investigated in an upcoming section.

4.3.2 Graph Learning Results

In order to better understand the clustering results, we now turn to the analyzing the view graphs. As was described in the methodology section, the graphs for all of the different views were created by a symmetric k-NN graph learning procedure. This procedure is meant to learn a graph that represents the data. So, graph-theoretic and network science measures can be used to analyze the graphs and thereby better understand the data. The following figures, Figure 4.3, display some important graph properties of the different view graphs for each of the daily data sets. The full numerical values of these measures for each day are available in the Appendices (Appendix B).

From the graphs, one can first observe that the graph densities follow a pattern that would be expected from the the number unique of hashtags. Density initially increases slightly, and then decreases as time moves forward. From the section on the data it is also easy to observe that this pattern is roughly the inverse pattern of the number of daily unique hashtags. So, as expected from observations of real-world networks, as the number of nodes — or unique hashtags, in this case — increases, the density decreases [94]. Additionally, the text view is consistently the most dense graph while the shared users graph is the least dense graph. Since these graphs were created by a symmetric k-NN graph learning procedure, most of difference in density for the users view is from more groups of overlaps, around the size of k, in users between hashtags. When there is more groups of overlaps around the size of k on the features, nearby objects in a k-NN graph tend to be within the k nearest neighbors of each other and that there are only about k neighbors for any given point, which results in fewer edges forming overall once the graph is symmetrized. Conversely, when there are more similar neighbors for each object than the value of k, there will be more edges in the symmetrized k-NN graph as two nearby objects, while very similar to each other, may not be within the top-k nearest neighbors of each other.

Second, the component statistics vary considerably between the different views of the data. The URLs view, which measures similarity between hashtags if they co-occur in tweets with the same URLs, has far more components than the other views, and these components — with the exception of one major connected component — are almost always isolates. This is to say that of the hashtags that co-occur with a URL, there are a fair number of URLs that only co-occur with a particular hashtag. This result is also partly an artifact of resolving the Twitter shortened URLs;

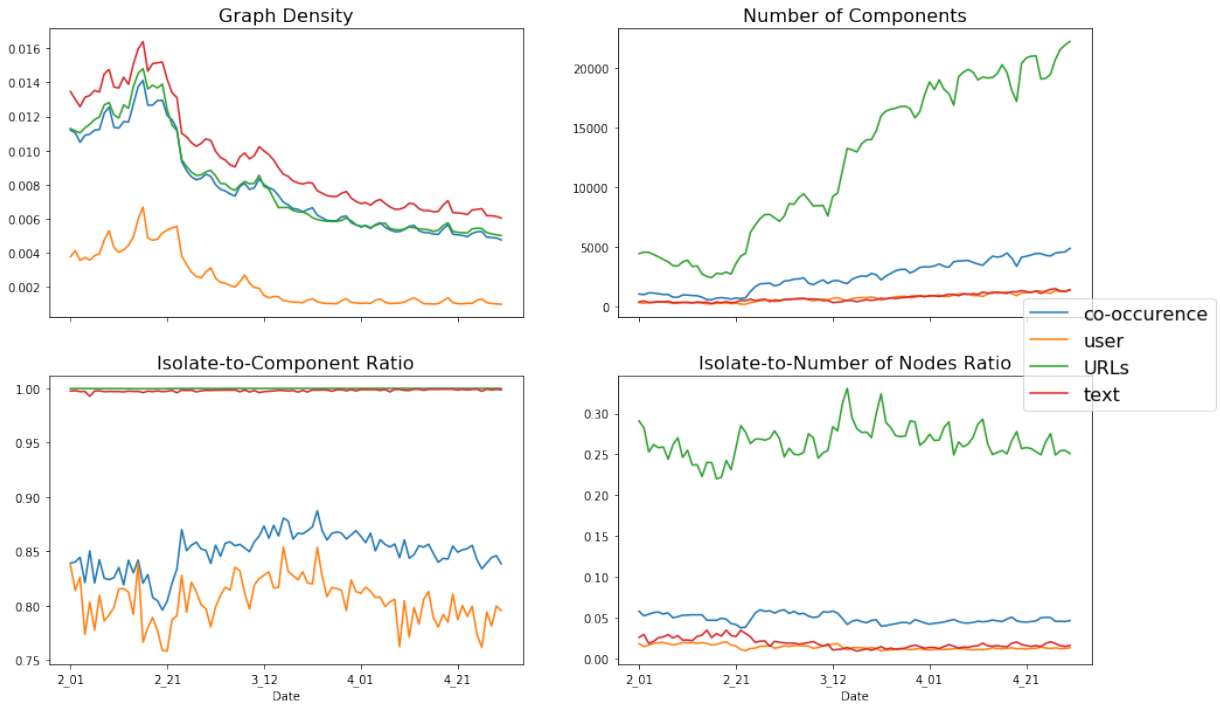


Figure 4.3: Graph metrics for each of the daily view graphs. All views but the URL view display useful graph properties for clustering the data.

some of the shortened URLs were unable to resolve to the non-Twitter URL, and Twitter does not always have the same shortened URL for any given URL. Thus, we would expect the URL view to not be particularly useful in finding communities of hashtag use. While much lower on the number of components, a similar pattern is observed with the text mode; those hashtags which are not part of the major connected component are almost always isolates. These hashtags are often rarely used hashtags, typically because they are a common misspelling of a popular hashtag or are a less popular hashtag that occurs with non-English text or no text (i.e. just the hashtag by itself was tweeted). Also, it should be noted that the co-occurrence view, which measures similarity between hashtags based on the other hashtags that those hashtags appear with in a tweet with, has a slightly higher number of components and percentage of isolates than either the user or text views. This is due to the fact that some hashtags never co-occur with another hashtag in a tweet. finally, it is worth noting that for all of the views, the number of components increase with time. These observations suggest that the use of hashtags is becoming more divided into distinct and non-interacting communities.

4.3.3 Multi-view Clustering Results

In this section, we turn to analyzing the performance of the MVMC technique on clustering the hashtags from the COVID-19 Twitter data. In the previous chapter it was established empirically that certain performance indicators of the MVMC method can give insight into the multi-view data itself, most notably the view weights and the number of iterations to reach (or not reach)

convergence. The following figures, Figure 4.4 displays the MVMC performance stats. The full numerical values of the performance stats for each view and each day are available in the appendices (Appendix B).

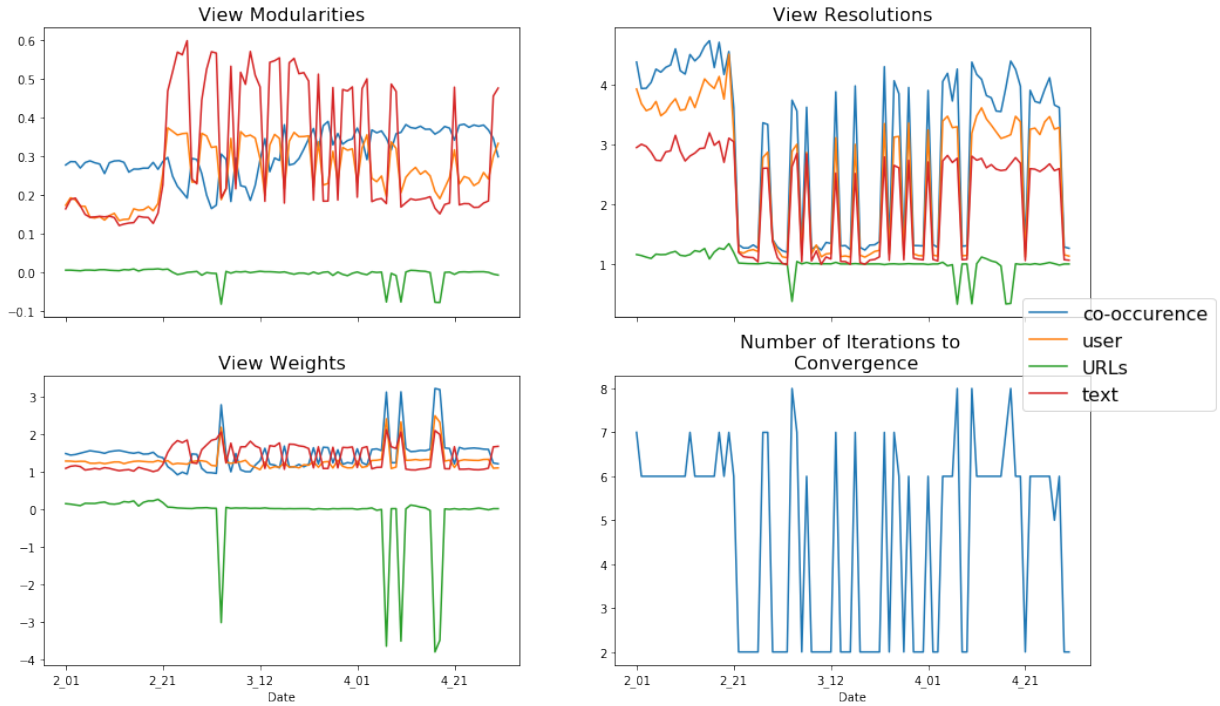


Figure 4.4: Performance indicators of the MVMC method for each day's clustering. Overall, the URL view was not useful in finding clusters in the hashtag data, while the other 3 views were roughly equally useful. The method also converged relatively easily indicating that there is cluster structure present among the hashtags, to varying degrees, within each day of the data.

The URL view does not provide much information towards clustering the data. The URL view is often close to zero, or even negative in terms of its modularity and view weights. The negative values occur on the 4th of March and the the 7th, 10th, 17th, and 18th of April. Its not clear why the URL view is especially poor at having a cluster structure these days, as these days are not distinct outliers in terms of the found clusters relative to any other day or in terms of the learned graphs on these days. It would seem that co-occurrence of a URL with certain hashtags did not reflect the co-occurrence of those hashtags in the other views and may indicate that the use of URLs, in a raw form, may not necessarily be a good indicator of a topical discussion community. The URL co-occurrence graph was especially sparse with many outliers, so this view likely would have benefited from some additional prepossessing, such as using the the top level domains of the URLs as opposed to trying to use the full, exact URL in determining similarity. Overall, as was hinted at in the previous section analyzing the learned graphs, the shared URLs do not contribute much to the cluster structure present within the hashtags.

As for the other views, they are all roughly equal in their contribution to clustering the hashtags, with some views being better on some days and others on other days. The view weights for the users, text, and co-occurrence of hashtags views are all generally between one and two and close to each other. As for the view resolutions, while the three useful views follow the same

temporal patterns, the co-occurrence consistently has a higher resolution followed by the user than the text views. This is due to consistent topological differences in the learned graphs, noted in the previous section. For example, the co-occurrence view is consistently more sparse than the text view and consistently has more components than either the text or the co-occurrence views. So, the other three views roughly contribute about the same in terms of clustering the hashtags.

Finally, there is a temporal pattern within the performance indicators of MVMC. From the 1st of February to the 21st of February all of the views obtain lower modularities and have higher resolutions and the MVMC method takes more iterations to to converge. These results indicate that this time period, which saw more and smaller clusters, was more difficult to cluster. From the 22nd of February through around the 4th of April, there is a lot oscillation in the view resolutions and modularities along with the number of iterations needed to reach convergence. So, this intermediary time period which had many shifts in the number and size of clusters also had variable performance in the ability of MVMC to cluster the data. Finally, from around the 5th of April to the 30th of April, view resolutions and the number of iterations required to reach convergence remain generally higher than the previous period and the co-occurrence view has consistently higher modularity than the other views. It was observed in the clustering results statistics that this same period also saw an increase in the number of clusters and a decrease in the size of the clusters. So, as with the other temporal periods in the performance of MVMC, when there are more numerous and smaller clusters within the data, it is more difficult to find clusters within the data and the clusters are less modular. Overall, MVMC never takes more than 8 iterations in order to reach convergence, which from previous empirical investigations of the technique, would indicate that there are cluster structures present within the multi-view data.

4.3.4 Temporal Ensemble Clustering Results

As has been noted throughout the results section and the data section, the usage of hashtags seems to have some temporal trends and changes over the period of investigation. So, in this section I will analyze the daily clusterings produced by MVMC to understand the temporal nature of the hashtag clusters and hashtag usage. To assess any possible temporal patterns that could exist within the daily clusterings, I analyzed how similar the daily clusterings are to each other. Comparing the similarities between the clusterings can give insight into how stable both the usage of hashtags and the broader discussion topics which use the hashtags are between days. In order to compare the daily clusterings, I opted to use the Adjusted Rand Index (ARI) [69], which provides a value between zero and one that expresses how similar two clusterings are. In order to measure the ARI between all of the daily clusterings, each clustering has to have the same objects, or hashtags, as every other clustering. So, a set of all of the hashtags used across the entire data set was collected from the filtered hashtag clusters (the filtered hashtag clusters are those clusters which have at least five hashtags in them, for each day). For each daily clustering, if a particular hashtag was not present on that day, it was added to the daily clustering and assigned a dummy cluster label. So, for each day, the clusterings have the same hashtags and those hashtags which do not occur on a particular day are all assigned the same dummy label for that day. Having cross-leveled the hashtags across all of the days in the data set, the pairwise ARI between each day's clustering and every other day's clustering can then be computed. The pairwise ARIs between the daily clusterings are summarized in the following figure, Figure 4.5.

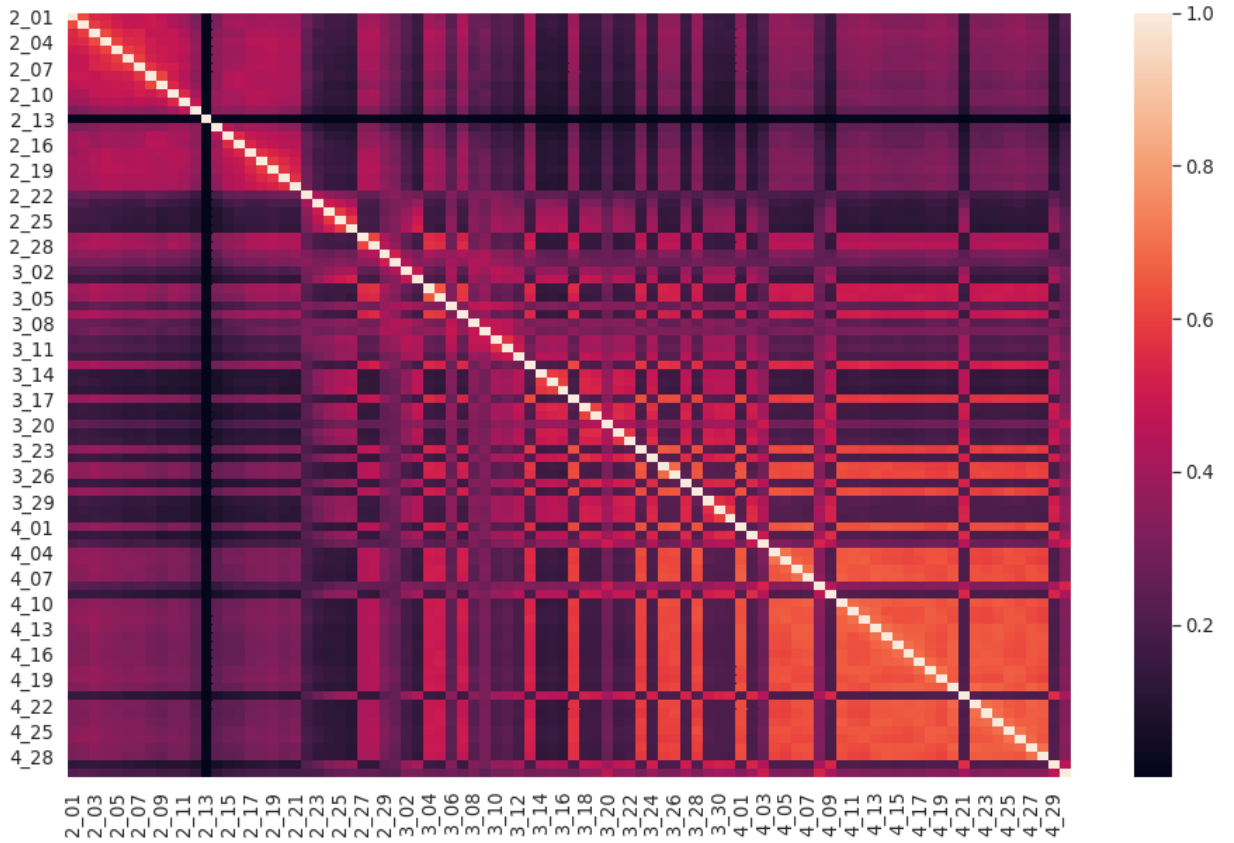


Figure 4.5: Heat map of the ARI values between every daily clustering produced by MVMC with every other daily clustering. The heat map shows there are some regions where the consecutive days are more similar to each other than to other days. Examples include early to mid-February in the top left and mid to late-April in the bottom right.

From the figure it can be observed that there are some block structures present within the data along with some outlier clusterings. For example, early to mid-February has clusterings which, with the exception of the 13 of February, are all more similar to each other than to any other days' clusterings. This temporal pattern was similarly observed in the clustering overview statistics and MVMC performance statistics. Additionally, clusterings in April also tend to display a block structure whereby the clusterings are more similar to each other than to any other days' clusterings. Outside of these block structures, there are also some outlier clusterings that are not more similar to those clustering which are temporally close. The 13th of February provides an extreme example in that it has very low similarity to every other clustering. Since it appears there are clusters of daily clusterings present within the data along with a pairwise measure of similarity, the clusterings can themselves be clustered. To cluster the daily clusterings, I used the pairwise ARI scores in Agglomerative Hierarchical Clustering with average linkage. It should be noted that clustering a set of clusterings has been used to analyze other temporal streams of data in order to understand dynamic trends within the data [87], [92]. The following the figure, Figure 4.6, displays the full dendrogram for the clustering of the daily clusterings.

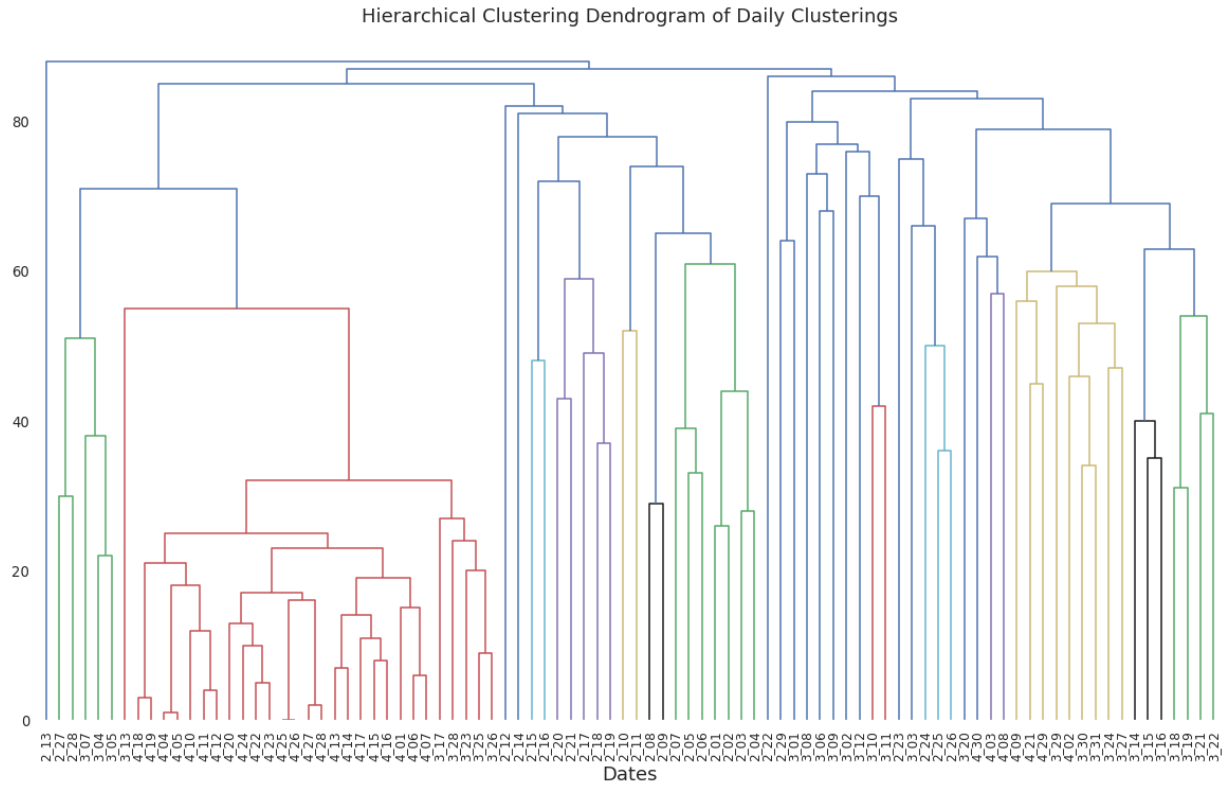


Figure 4.6: Cluster Dendrogram of the daily clusterings produced by Agglomerative Hierarchical Clustering with ARI as the measure of similarity. The Dendrogram shows 3 main clusters with some outlier daily clusterings.

From the dendrogram it can be observed that there are indeed clusters of the daily clusterings. And, these clusters tend to consist of temporally nearby clusterings. Thus, it would seem that there are temporal meta-clusters of daily clusterings present in the data. To analyze these temporal meta-clusters, I first clustered the clusterings. Based the dendrogram, the clusterings were divided into 5 clusters. Note that the division of the daily clusterings is done without respect to time, but rather is done only on the pairwise ARI between the daily clusterings. Having partitioned the daily clusterings into meta-clusters, we can then see if these meta-clusters correspond to any time periods within the data. The following figure, Figure 4.7, displays a plot of the daily clusterings over time versus the meta-cluster that the daily clusterings were partitioned into.

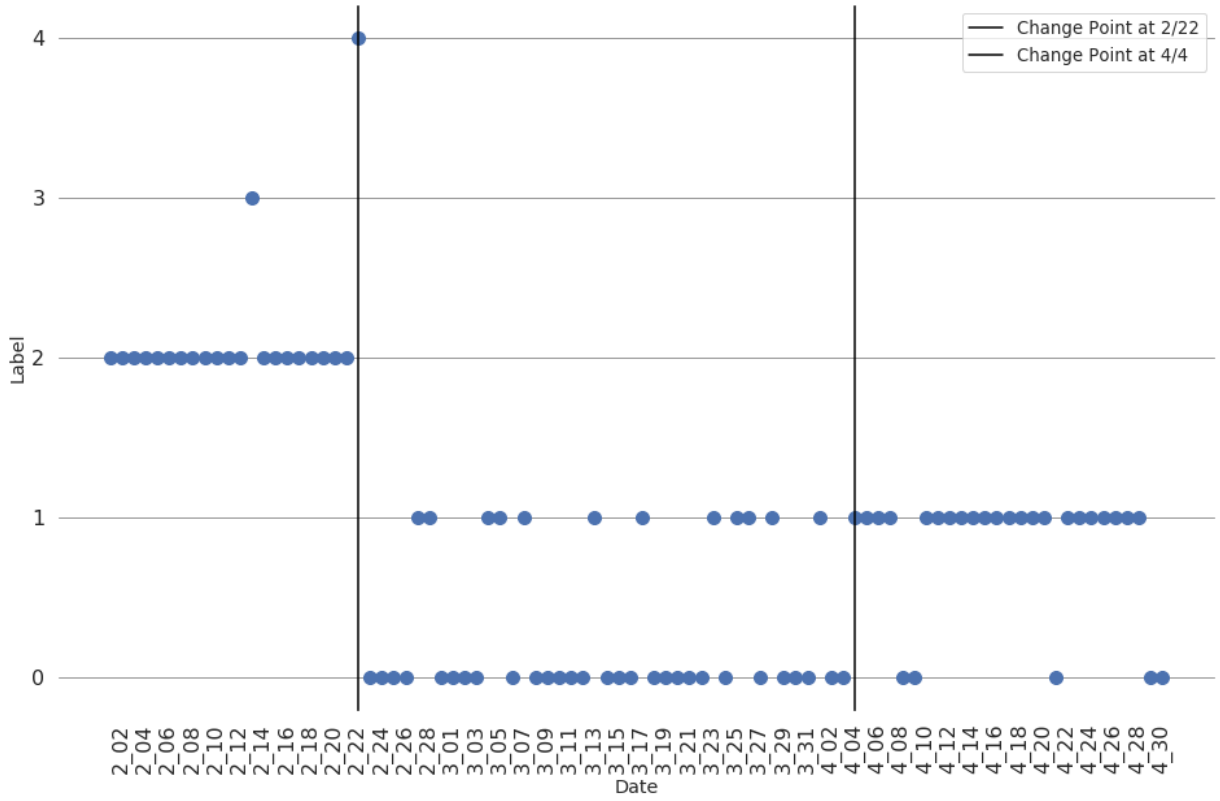


Figure 4.7: Plot of daily clustering versus what meta-cluster that daily clustering falls into. Generally, the meta-clusters of the daily clusterings have distinct temporal bounds indicating that daily clusterings have a macro, temporal structure to them.

From the figure, there is an observable temporal pattern to the meta-clusters. There is a meta-cluster (label 2) that is exclusively composed of the clusterings from early to late-February. The other two meta-clusters largely contain clusterings from either February 23rd to April 3rd, or April 4th to April 30th. There are also two outlier meta-clusters that consist of only one date, February 13th and February 22nd (labels 3 and 4, respectively). Thus, there is a macro temporal pattern within the daily hashtag clusterings. This temporal pattern in the hashtag clusterings reflects a similar pattern regarding user hashtag ratios that were observed in the data section. In fact the middle period of clusterings, which is the least distinct of the 3 time periods — having meta-cluster 0 and 1 members — corresponds closely to the time period where there were many oscillations in MVMC performance. So, not only does hashtags usage on the user level change over the course of the pandemic, but also the topical groups of hashtags also change over the course of the pandemic. And, it would seem there are two stable periods of hashtag usage that occur during the early stages of the pandemic in February and after the pandemic had been raging globally for some time, in early April.

4.3.5 Analysis of Temporally-Ensembled Clusters

Having observed three distinct time periods of hashtag clusters, we would now like to get a better sense of how these periods differ in terms of hashtag usage. In order to better understand

the hashtag clusters from the different time periods, each of the clusterings making up a meta-clustering are transformed into one clustering through cluster ensembling. This is done for two reasons: First, it makes the selection of which days and which clusters to analyze less arbitrary by reducing the number of clusters that need to be analyzed. Second, producing an ensemble clustering for each of the time periods can better mitigate any daily idiosyncrasies that could affect any given clustering on any given day, and thereby produce a better overall clustering that represents the whole time period. To produce ensemble clusters for each time period the BGPA+ technique, which clusters the object-by-cluster graph, was used. This technique was used as it displayed consistently strong performance among the many cluster ensembling techniques tested in Chapter 2 and it is a very scalable technique. The following table, Table 4.1, displays a summary of the results for each of the ensembled period meta-clusters.

Period	Number of Clusters	Average Size of Clusters	STD Size of Clusters	Average Internal ARI
February 1 - February 22	14	751	1104	0.416
February 23 - April 3	13	747	1093	0.348
April 4 - April 30	16	566	691	0.536

Table 4.1: Cluster statistics of the ensembled clusterings of the daily clusterings for each time period. The ensembled clusterings display similar temporal trends to the other analyses of the data in that attributes like the number of clusters present in each period roughly reflect the numbers of daily clusterings and unique hashtag usage during these periods.

The ensembled, period meta-clusters produced somewhat different cluster structures for the different time periods. The third period had slightly more clusters and smaller and more regularly sized clusters than the other time periods. This time period also had a a higher clustering similarity, in terms of the average pairwise ARI between its constituent daily clusterings. For the first and second time periods, there was often one large cluster that had a size of around 4,400 hashtags while the largest cluster in the third time period was 3,025 hashtags. So, the third time period has a more balanced cluster structure, across the entire time period, than the other two time periods.

To get a better idea of the ensembled clusters, the following table, Table 4.2 provides a qualitative assessment of the topic of cluster as well as how focused and easily assignable a topic is to each of the clusters in each of the time periods.

Period 1	General Topic	Focus Level of Cluster
0	Multi-language, general use hashtags	low
1	Multi-lingual coronavirus-specific hashtags	low
2	News resources related hashtags	medium
3	Chinese focused hashtags (often of negative sentiment)	high
4	Thai related hashtags	medium
5	Economy/Commerce related hashtags	high
6	U.S. Politics related hashtags	high
7	technology and business related hashtags	high
8	Asian-languages hashtags	high

9	Multi-lingual, anti-racism and health news related hashtag	medium
10	French language and European related hashtags	medium
11	Italian language hashtags	high
12	Arabic script hashtags	high
13	All-caps hashtags with some Syrian Civil War hashtags	medium
Period 2	General Topic	Focus Level of Cluster
0	News and U.S. Politics related hashtags	medium
1	News-resources related hashtags	low
2	Commerce, Economy, and technology related hashtags	medium
3	Asian-languages hashtags	medium
4	Spanish language hashtags	medium
5	French language hashtags	high
6	Italian language hashtags	high
7	Turkish language with some conspiracy theory related hashtags	medium
8	Online entertianment related hashtags	hmedium
9	German language hashtags	medium
10	Arabic script and middle-east related hashtags	high
11	Australian and British news related hashtags	high
12	Online education related hashtags	high
Period 3	General Topic	Focus Level of Cluster
0	Spanish language and many general hashtags	low
1	Multi-lingual, coronavirus and general hashtags	low
2	Commerce, Economy, and technology related hashtags	medium
3	Multi-lingual, coronavirus and general hashtags	low
4	British news, anti-racism, and medical science related and hashtags	medium
5	U.S. Politics related hashtags	high
6	Arabic-script and location related hashtags	medium
7	Chinese focused hashtags (often of negative sentiment)	high
8	French language hashtags	high
9	Italian language hashtags	high
10	Canadian and climate change related hashtags	high
11	German language hashtags	medium
12	Asian languages hashtags	high
13	Indonesian and surrounding countries related hashtags	medium
14	Thai language hashtags	high
15	Turkish language hashtags	high

Table 4.2: Topical labels for the temporally-ensembled meta-clusters. Some clusters had much more focused and readily defined topics than did others. Also, some topics are persistent throughout all three time periods, while some only exist in a time period.

Looking at the hashtags present in the clusters of the different ensembled meta-clusterings revealed both persistent topical groups and ones which change over time. In every time period, there is always a cluster that has hashtags for breaking news or news sources, a cluster that has business and commerce related hashtags, a cluster that has U.S. Politics-related hashtags, and

foreign language clusters — most notably Italian, German, and Spanish. These topical groups indicate that conversations about the global economy, news, and U.S. politics have remained important and consistent topics throughout the pandemic, and that even with English-language collection terms, the discussions occurring around the COVID-19 pandemic are international in nature. In addition to these persistent clusters there are also transient cluster topics that emerge in some time periods but not others. For example, in time period two there is a cluster of hashtags dedicated to online education and a cluster of hashtags concerning online entertainment and entertainment services (i.e. Hulu, Netflix). Both first and third periods contain clusters with negative sentiment hashtags toward the Chinese government and in support of Hong Kong protests. Overall, there are consistent topical clusters of discussion and other topics which emerge and disappear over time.

4.3.6 User-base Analysis of Temporally Ensembled Clusters

In order to get a better sense of the hashtag clusters found through multi-view clustering and temporal ensembling, we can analyze the users that use the hashtags. In particular, it is of interest to observe whether those individuals which most use a hashtag also frequently use other hashtags from the same cluster. Presence of a small number of users being most active in the use of the hashtags could give insight into whether the topical conversation is being driven by a small group of users or is more of an open, less centrally-dominated topical discussion. To do so, I first found the top third of users for each hashtag in each cluster, across all periods, which I refer to as the 'top users.' I then analyzed the number of unique top users for each cluster in each time period. The number of unique top users within any given topical hashtag group can give insight into whether there is a diverse user-base driving the topical discussion or not. The following figure, Figure 4.8, shows the number of unique top users for each cluster in each time period.

From the figure it can be observed that there are differences in user bases both between time periods and between clusters. The first and third time periods have generally fewer unique users in each of their clusters than the second time period. This due in a large part to the previous observation that there are more unique users in general on any given day during the second period than there are for the first or third periods. It is worth noting, however, that the total ratio of unique users to total users in both the second and third periods are about the same at 0.202 and 0.206 respectively. That is to say that even as the number of unique users decreases slightly in the third period and that the clusters in the third period do not have as many unique users, the period retains a relatively high number of unique users across the time period. Additionally, there are distinct differences in the number of unique top users between clusters within any of the time periods. This is especially true for the second time period. Generally, this difference in unique users is only partly accounted for by a difference in the size of the clusters as the 0th and 1st clusters are always the largest in any given period but do not have the greatest number of unique users for those time periods. This discrepancy in numbers of unique top users is also a result of the generality of the particular topic of the clusters, with those clusters having more general topics having higher numbers of unique top users. For example, the 4th cluster in the second period contains many hashtags, from many languages, which describe COVID-19, such as '#COVID-19', '#covid19', or '#COVID—19.' So, the topical clusters of hashtags have

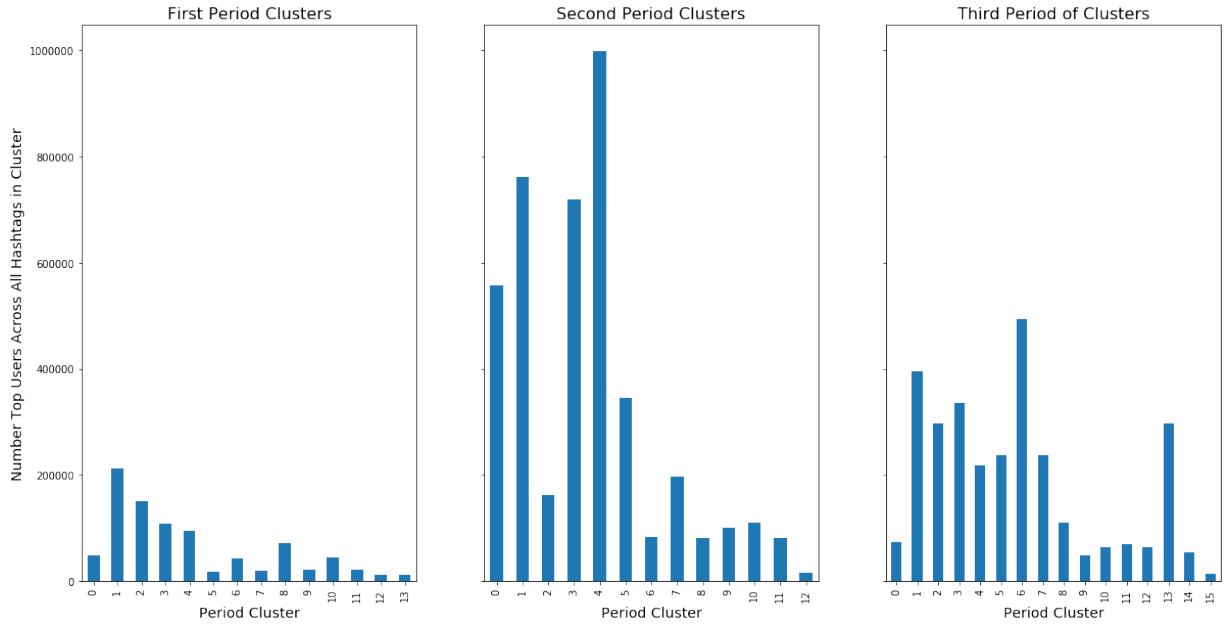


Figure 4.8: Number of unique users for each cluster of hashtags in each time period. There are distinct differences between the number of unique users between and within periods. These differences are largely not driven by the number of hashtags present within a cluster, but rather with the topic of those hashtags.

differences in their top users with some clusters having a very small top user base and others, a larger one.

One of the issues with just looking at the the number of unique top users is that hashtags have different numbers of users in general. So, a hashtag could be completely used by a different user in each use, but the hashtag itself is not widely used, which would result in that hashtag having a small unique top user base. This effect extends to clusters where there are clusters of generally less used hashtags. So, I also use a *top user score* for each cluster which compares the ratio of the number of top unique users for the hashtags versus the number of top unique users if there was no overlap between the top unique users of the hashtags. As a mathematical expression, this top user score for a cluster is given by:

$$\text{top user score}^c = \frac{\sum \bigcap_{i=1}^{r} top\_users_i^c}{\sum_{i=1}^{r} \sum top\_users_i^c} \tag{4.3}$$

where c is a particular cluster, r is the number of hashtags present in cluster c, and $top\_users_i^c$ is the set of users for hashtag i in cluster c. The following figure, Figure 4.9, displays the top user scores for each of the clusters in each of the time periods.

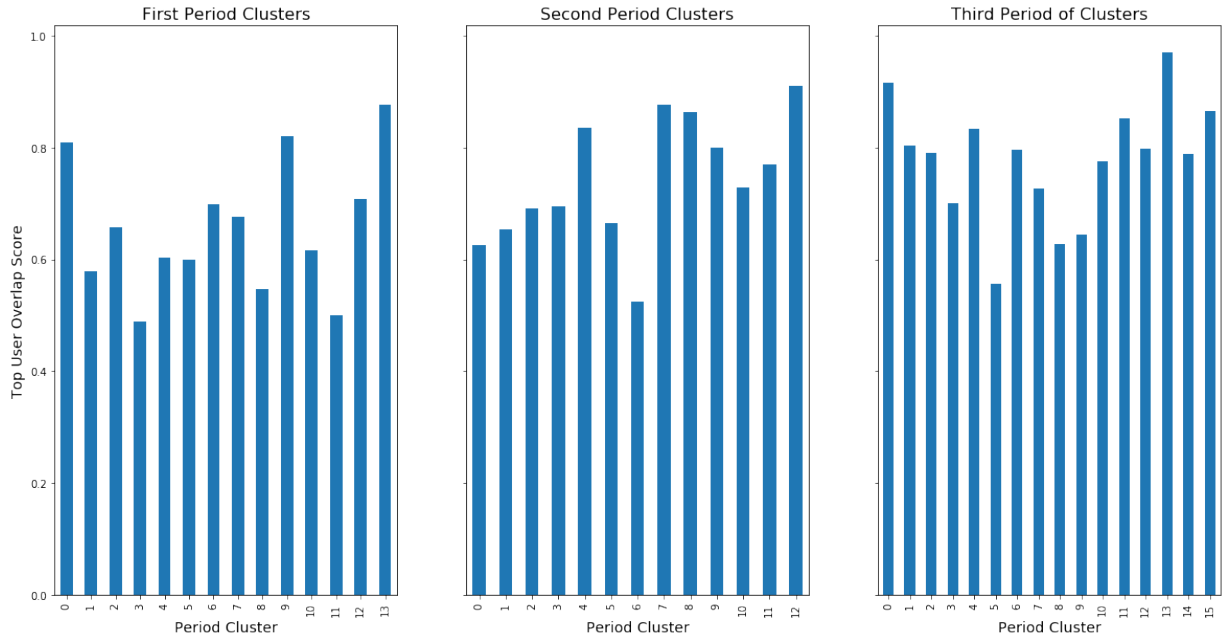


Figure 4.9: Top user scores, which compares the number of unique users for each of clusters to what would be expected if there were no overlap in the unique users between hashtags from the same cluster. Using this normalized measures allows us to observer differences in diversity in user bases of clusters which may have a small, but diverse, number of users

From the figure, normalizing the unique top users by the actual hashtag usage across the clusters produces some different results than the previous figures. Some of the smaller clusters with a small user base can actually have a very diverse user base. For example, period one cluster 13, which focuses around the Syrian Civil War, has the highest top user score but a relatively small number of unique users. Another example is period 2 cluster 12, which focuses exclusively on online education-related hashtags, has a small number of unique users but the most diverse user base for the second time period. So, some clusters which can be small in the number of users can have very different users using the hashtags. This observation would suggest that only analyzing hashtags by creating hashtag-to-hashtag networks based on users is actually insufficient to find clusters of hashtags, which was a similar result observed by other authors [131]. Additionally, some of the more mid-sized clusters can have less diverse user bases. For example, period one cluster 3, which focuses on hashtags critical of the Chinese government, has the least diverse user base in the first period, but a fair number of unique users. Period two cluster 6, which has many Italian-language hashtags, and period 3 cluster 5, which focuses on U.S. politics, have similar patterns. So, the topic of of a cluster tends to drive how diverse the user base of that cluster is, and not the number of hashtags or even the number of unique users.

To further explore the nature of the unique users of hashtags within clusters, we can look at how unique each hashtag's user base is within each cluster. In order to better understand the user base of a particular hashtag I calculated the ratio of the number of unique users that use a hashtag versus the number of times a hashtag is used in a given period. These hashtag user scores can then be combined to analyze each cluster within each of the time periods. The following figure, Figure 4.10, displays a box and whisker plot of the hashtag user scores for each of the cluster for

each of the time periods.

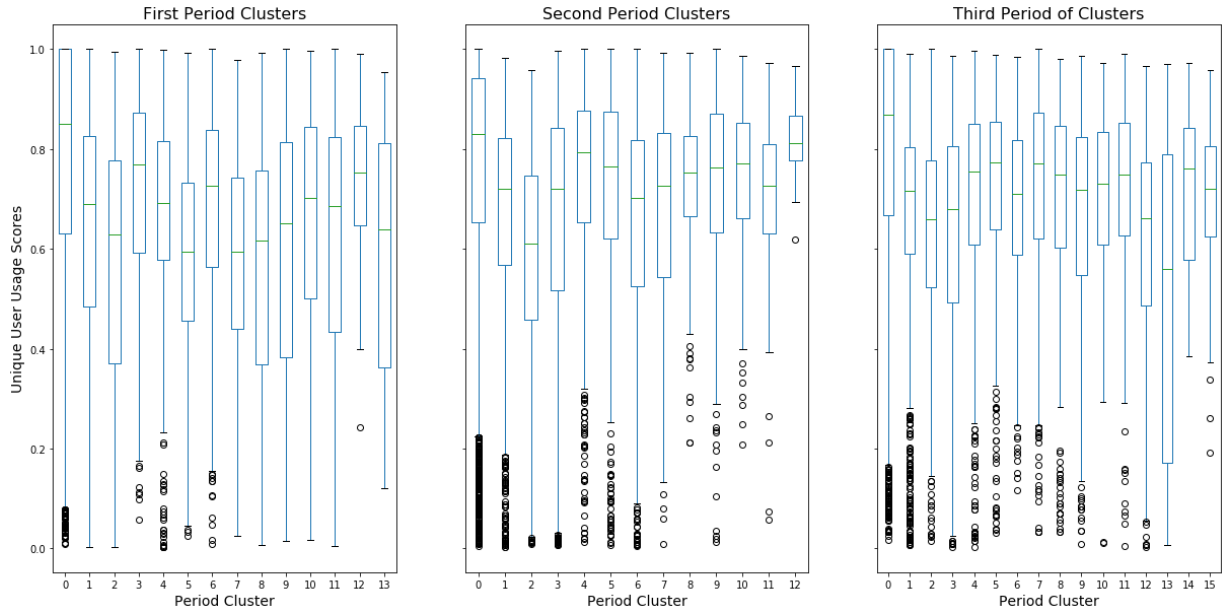


Figure 4.10: Box and whisker plot of the unique user scores for each of the hashtags within a cluster. Generally, each cluster has a relatively wide spread of scores which indicates hashtags that have many unique users and those with very few being present in each cluster. Many clusters also contain low-scoring outliers which are hashtags that are used by very few or even just one user.

The unique user scores for each of the hashtags within each of the clusters show distinct differences between the clusters within each period. First, it is worth noting that the spread of scores for the different clusters tend to be consistently wide; most clusters have scores ranging from 1.0 to near 0.2. There are notable exceptions to this, like period 2 cluster 12 which focuses on online education, and period 1 cluster 12 which is an Italian-language cluster. Second, while there is a wide range of scores for the hashtags within any given cluster, most clusters often have outlier hashtags with very low user scores. That is to say, that most clusters have at least one hashtag that is often tweeted, but only by a few or one user. So, while the clusters often present distinct themes in their hashtags, the usage of the hashtags within the clusters can vary considerably. Some hashtags have a much more diverse base of usage while some are promulgated by only a few users. This result once again suggests that clusters of hashtags form around topical groups and that all hashtags are not equal, at least in terms of usage, within a topical group.

Overall, the user base of the different hashtag clusters demonstrate that not only are the clusters very different in their user make up, but so to are the user bases for individual hashtags within the clusters. Certain topical areas of hashtag usage, and by extension, discussion, tend to be driven by a small number of users, whereas others have a much more diverse user base. Furthermore, within clusters of hashtags, hashtags can vary quite a lot with their user bases, with some hashtags being promulgated by very few, or even one, users. So, even within a certain topical area of hashtags, a small group of users will control the usage of certain hashtags and possibly parts of the discussion as well.

4.3.7 Detailed Analyses of Particular Clusters

In this section, I analyze a few of the clusters more deeply. In particular, I analyze the usage statistics of different hashtags within the clusters as well as the verbiage associated with those clusters.

First Period, Chinese-focused Cluster

The first cluster of interest is a Chinese-focused cluster from the first period. This cluster was composed almost entirely of hashtags relating to either China or Wuhan. It should also be noted that this time period had a hashtag that featured the term 'china' or 'wuhan' in every single cluster except the cluster focused around business and commerce (period 1, cluster 11) and the cluster focused around U.S. Politics (period 1 cluster 6); around 86% of the clusters in the first period featured a hashtag with one of these terms. In contrast, 68% of the clusters in the second period and 50% of the clusters in the third period had a hashtag with one of these terms. The following table, Table 4.3, displays a sample of some of the hashtags present within the cluster. Additional examples of the hashtags from this cluster are available in the appendices (Table B.4 in Appendix B).

Period 1, cluster 3: Chinese-Focused					
Most Used Hashtags	Number of Uses	Hashtag with Highest Original User Ratio	User Ratio	Hashtags with Lowest User Ratio	User Ratio
WuhanVirus	145252	WeStandWithHongKong	1.000	myedgeprop	0.058
HongKong	116401	HKpolicestate	1.000	hongkonggenocide	0.108
CCP	52252	Catastrophy	1.000	usdjpy	0.112
Hubei	46059	Darkness	1.000	china_is_territorist	0.122
Chinese	44336	ProtestArt	1.000	TechJunkieNews	0.161
WuhanPneumonia	31435	HKexit	0.998	EiSamay	0.165
Coronarivus	31152	Hubie	0.995	hongkongprotest	0.176
WuhanCoronavirusOutbreak	26612	timelapse	0.985	CaptainTripps	0.196
LiWenliang	26352	stayclam	0.982	Análisis	0.202
HongKongProtests	26324	PLAAF	0.982	EnvironmentHealth	0.207

Table 4.3: Top raked hashtags from period 1, cluster 3 which is Chinese-focused in its hashtags. Many of the hashtags from this cluster are critical of the Chinese government for either its response to recent protests in Hong Kong or for its response to the COVID-19 outbreak from Wuhan.

In general, the hashtags within this cluster express negative sentiment towards China and the Chinese government. Some of the most used hashtags within the cluster express support for Hong Kong protests and symbols of frustration with the Chinese government, like '#LiWenliang' [35]. This trend is further emphasized with both those hashtags which are used by a diverse user base (highest original user ratio) and those used by very few users. Overall, there is a mix of anti-Chinese hashtags and pro-Hong Kong and pro-Tibet hashtags present within the cluster. So, in the first period, which are the early days of the COVID-19 pandemic there is a distinct vein of discussion within the COVID-19 discussions that is expressing negative sentiment toward the Chinese and Chinese government. It is interesting to observe that this negative sentiment is not limited to COVID-19 but also encompasses other issues with the Chinese government to include protests in Hong Kong.

To get a better idea of nature of the discussion that uses these hashtags, we now turn to the text view and the words that co-occur with the hashtags in the cluster. The following figure, Figure 4.11 displays a word map of the common words that co-occur with hashtags from the cluster.

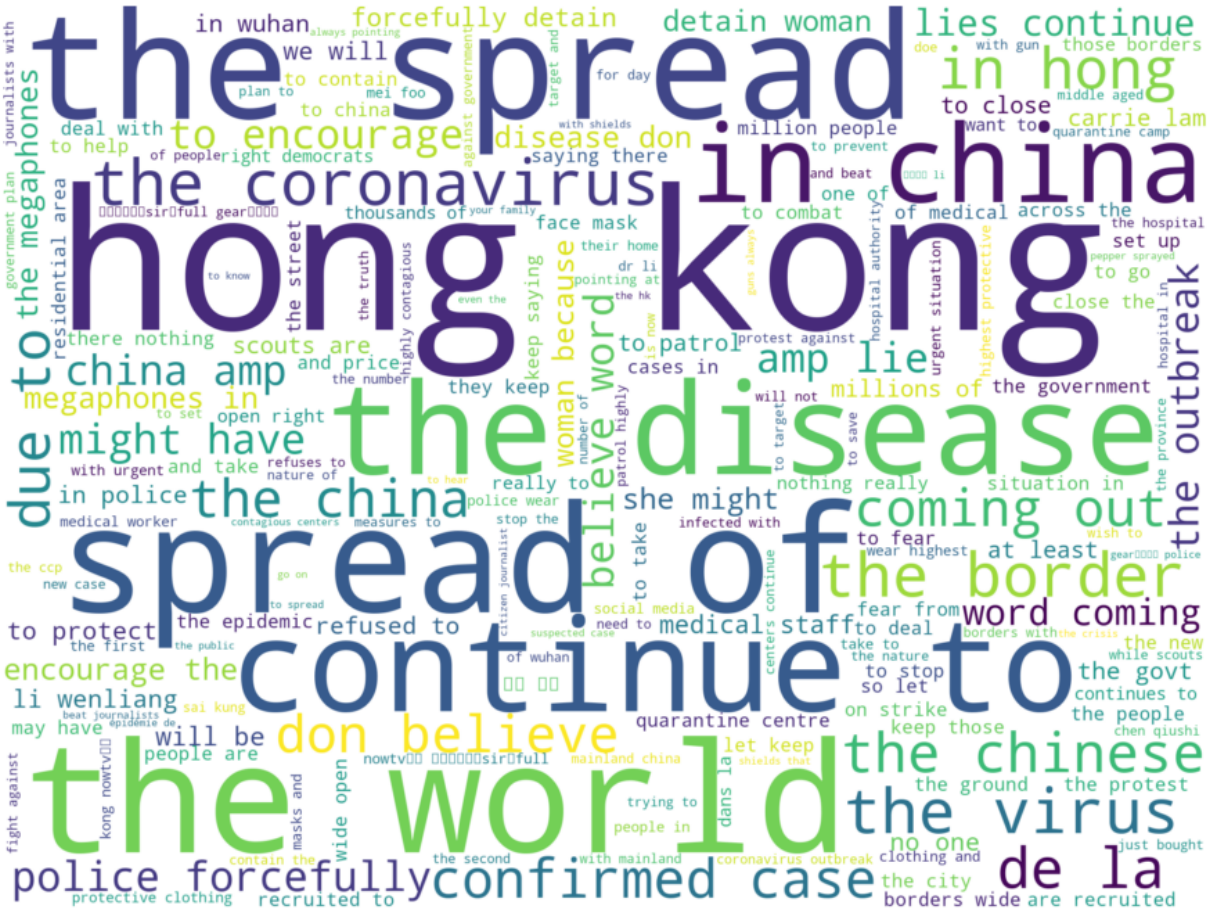


Figure 4.11: Word map of frequently occurring words and phrases that co-occur with hashtags from the Chinese-focused cluster in the first period. There is a significant amount of verbiage about the spread of the virus and about protests in Hong Kong.

As with the hashtags themselves, the words that often co-occur with the hashtags indicate a focus on the spread of the Coronavirus and Hong Kong related issues. Thus, it would seem that in the early period of the data there is a significant amount of negative discussion surrounding China and the Chinese government both for recent actions in Hong Kong and for being the source of the COVID-19 pandemic. When this result is combined with the user base analysis results of this cluster which indicate that this cluster has the most overlap between the users of its hashtags, it would seem that the discussion and condemnation of Chinese government actions during the early part of the pandemic is driven by relatively few users. This result also indicates that those users who were already critical of the either the Chinese response to Hong Kong protests or to the initial Chinese government reaction to the COVID-19 pandemic may be using the other calamity to draw attention to their calamity of focus. So, while Chinese related terms feature prominently

in many of the clusters of hashtags in this period, there is also a distinct cluster of hashtag usage that is critical of the Chinese government.

First Period, Syrian Civil War Cluster

The next cluster of interest is a cluster that features content centered on the Syrian Civil War. At first glance, this is already a strange cluster to have in a data set that was collected based on COVID-19 tweets; there is not an obvious connection between the two entities beside the fact that they are both significant, contemporary calamities. Additionally, the cluster has a high user ratio score, meaning different user accounts are using different hashtags. So, it would seem there is actually a diverse base of users supporting the different hashtags, but the content is almost solely focused on the Syrian Civil War. The following table, Table 4.4, displays some of the salient hashtags from the cluster.

Period 1, cluster 13: Syrian Civil War					
Most Used Hashtags	Number of Uses	Hashtag with Highest Original User Ratio	User Ratio	Hashtags with Lowest User Ratio	User Ratio
CORONAVIRUS	38642	BILLGATES	0.955	AssadGenocide	0.122
CHINA	10509	Nation	0.944	Assad_Torture	0.123
WUHAN	2397	ACTUALIZACIÓN	0.929	Chemical_Assad	0.123
NCOV19	1543	LAMORGESE	0.927	TheResistance1776	0.123
IndianArmy	1316	FAKENEWS	0.908	AssadCrimes	0.123
ALERT	1294	CINA	0.899	PutinAtWar	0.124
BIOWEAPON	992	SINGAPORE	0.896	WhiteHelmets	0.125
Syrie	981	BIOWEAPON	0.860	InfoWars	0.215
VIRUS	910	BEIJING	0.859	TBT	0.233
AssadGenocide	831	ALERT	0.850	VIRUSCORONA	0.241

Table 4.4: Hashtags from period 1 cluster 13, which has content and some hashtags devoted to the Syrian Civil War. Many of the hashtags within the cluster have non-overlapping users and are often all caps versions of more well known hashtags.

The hashtags present in this cluster differ from hashtags in other clusters. For one, many of the hashtags in use are all caps versions of other hashtags, such as '#CORONAVIRUS' for '#coronavirus'. Second, those hashtags which have very little overlap are almost all of the all caps variety while those with much less user overlap are more widely used and recognized hashtags, like '#WhiteHelmets'. To get a better idea of the usage of the hashtags present in this cluster, the text co-occurring with the hashtags can be analyzed. The following figure, Figure 4.12, displays a word map for the commonly used text that co-occurs with the hashtags in this cluster.

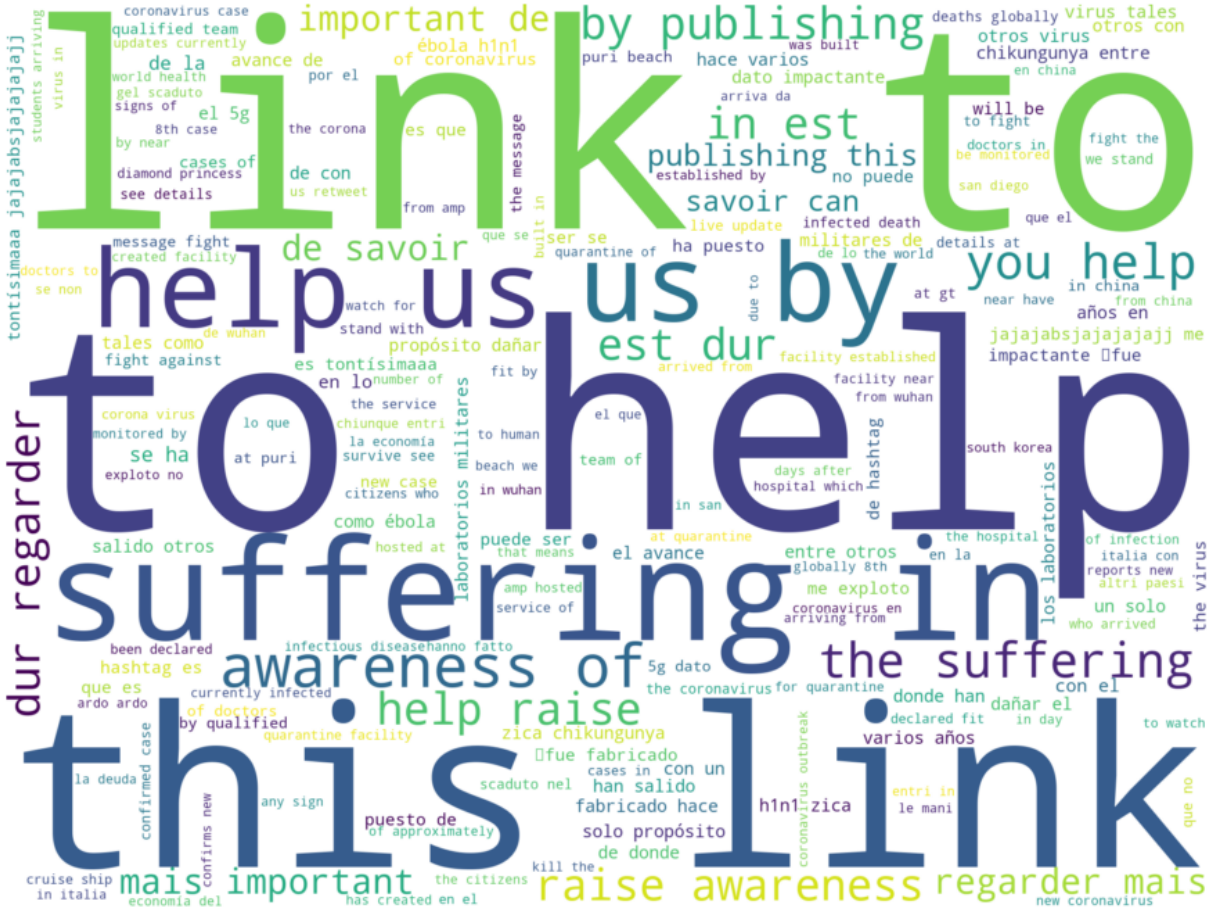


Figure 4.12: Word map of commonly occurring phrases and words from period 1, cluster 13. Much of the verbiage seems aimed at directing users to links to support various parties involved in the Syrian Civil War.

As with the hashtag themselves, the text co-occuring with the hashtags is different from the other clusters. Phrases like ‘link to’ and ‘to help’ feature prominently in the accompanying text and the accompanying text is much more focused in the primary topical area of the cluster than other clusters. Additionally, the text is multi-lingual in that there are mostly English words, but also words from Spanish, French, and others. So, from the text, these hashtags are used to promote awareness of the Syrian Civil War and to draw users to websites to provide support to various organizations involved in the Syrian Civil War. Thus, these hashtags seem to be aimed at using the COVID-19 pandemic to draw user interest to the Syrian Civil War. Additionally, it is also interesting to note that while the text is very uniform in its drawing attention to the Syrian Civil War, the user base is actually very diverse. This cluster has the most diverse user base of any cluster in the first time period. So, these results would suggest that while there is a diverse user base for this cluster, that this diversity is artificial; the users that use the hashtags within this cluster are probably related and possibly being centrally coordinated. At any rate, it is clear from the results that the hashtag usage in this cluster is meant to use the COVID-19 pandemic to draw attention to an entirely different calamity.

Second Period, Online Education Cluster

Another cluster of interest is a cluster of hashtags that only exists in the second time period and focuses exclusively on online education-related hashtags. This cluster contains relatively few hashtags (37 in total) that all relate to online learning. It also exists only in the second period when most of the world entered some form of lock-down to slow the spread of the coronavirus. Despite the few number of hashtags in the cluster, this cluster has the most diverse user base in the second period and has all of its hashtags generally having a diverse usage (the lowest user score for a hashtag in the cluster is 0.619). The following table, Table 4.5, displays the some of the salient hashtags from the cluster.

Period 2, cluster 13: Online Education					
Most Used Hashtags	Number of Uses	Hashtag with Highest Original User Ratio	User Ratio	Hashtags with Lowest User Ratio	User Ratio
education	9817	child	0.967	Education	0.619
onlinelearning	4386	teaching	0.907	Learning	0.694
edtech	4673	AcademicChatter	0.907	STEMeducation	0.712
college	3530	college	0.901	EdChat	0.714
AcademicTwitter	3606	student	0.899	intled	0.734
distancelearning	3505	virtuallearning	0.886	highered	0.739
AcademicChatter	3204	universities	0.879	edtech	0.751
edchat	3594	AcademicTwitter	0.876	HigherEd	0.761
online	3235	students	0.874	university	0.774
STEM	2936	distancelearning	0.867	education	0.777

Table 4.5: Hashtags from period two cluster 12 which focus on online education. The hashtags used in this cluster have a diverse user base and are focused in that there are no hashtags that are not easily identifiable as being education-related in the cluster.

As was mentioned previously, all of the hashtags in this cluster relate to education, especially online education, and have diverse user bases. In order to better understand the nature of the usage of the hashtags we turn to the words that co-occur with these hashtags. The following figure, Figure 4.13, displays a word map for the frequently used words and phrases from the cluster.

Figure 4.13: Word map for commonly used words and phrases present in cluster 12 from period 2, which has education hashtags. The verbiage is a mix of primarily English and French and has many phrases relating to school closures and new school hygiene policies.

Despite the hashtags all being in English, there is a surprising amount foreign language text, especially French, that co-occurs with these hashtags. Much of the text relates to the closing of schools, and new health-related procedures for schools. These hashtags were used not only to promote online learning solutions and products but also to advertise moving of courses to online and new school-related health procedures. So, this cluster highlights the use of hashtags in order to inform groups of users about both relevant happenings as well as education alternatives during a pandemic. In that sense, this cluster differs from many of the other clusters in that it focuses note only on different content but also seems more aimed at a sharing of non-partisan information to users.

Comparison of U.S. Politics Focused Clusters

Finally, the U.S. politics focused clusters from periods one and three are analyzed to both understand their content and how the discussion topic of U.S. politics has changed over the course of the COVID-19 pandemic. In the first period, the U.S. politics focused cluster is slightly above average in terms of the user base diversity and has a relatively large number of hashtags at 343. The following table, Table 4.6 displays the salient hashtags from the cluster. Additional examples of the hashtags from both time periods U.S. politics related clusters are available in the appendices (Table B.5 in Appendix B).

Period 1, cluster 6: U.S. Politics					
Most Used Hashtags	Number of Uses	Hashtag with Highest Original User Ratio	User Ratio	Hashtags with Lowest User Ratio	User Ratio
MAGA	13842	OpenBorders	1.000	hillaryemails	0.008
Trump	11976	Newyork	0.999	HAction	0.018
FakeNews	9905	TheGreatAwakeing	0.994	StopTheMadness	0.048
AmericaFirst	9150	DemCast	0.985	bluelivesmatter	0.062
QAnon	8456	ThesePeopleAreSick	0.982	ImpeachTrump	0.105
GatesFoundation	7768	GatesFoundation	0.981	TRoom	0.106
Dobbs	7752	IngrahamAngle	0.980	ABQ	0.135
Newyork	7131	TrustThePlan	0.960	rockoftalk	0.135
FoxNews	6209	VoteBlueToEndThisNightmare	0.959	NM	0.140
FreeZeroHedge	5736	JoeBiden	0.955	Galaxy	0.148

Table 4.6: Hashtags from period one cluster 6 which focuses on U.S. Politics related clusters. There is a mix of hashtags associated with political news, political personalities and prominent politically-based conspiracy theories. The hashtags from this cluster also have a wide range of user bases in terms of the uniqueness of the users that use the hashtags

The hashtags are a mix of political commentary, hashtags related to prominent political figures, and hashtags typically related to conspiracy theories (i.e. '#QAnon'). Some of the most used hashtags within the cluster relate directly to current U.S. President Donald Trump (i.e. '#MAGA' and '#Trump') while some of the most diverse hashtags in the cluster are anti-President Trump (i.e. '#VoteBlueToEndThisNightmare' and '#DemCast'). It is also interesting to note that the two hashtags with the least diverse user base — which have scores well outside the inter-quartile range of user scores for the cluster — have only one user who posts both hashtags, 119 and 56 times respectively. So, the cluster contains various hashtags related to various elements of U.S. Politics, including those which are generally associated with partisan content. Thus, in some respects, as with the Syrian War Cluster this cluster contains hashtags which are attempting to use the COVID-19 pandemic in order to draw attention to certain political views or ideas.

To get a better idea of the hashtag usage in the first period's U.S. Politics cluster, I analyzed the text which co-occurs with the hashtags. As with the previous analyses of other clusters, the following figure, Figure 4.14, displays the word map for the co-occurring text.

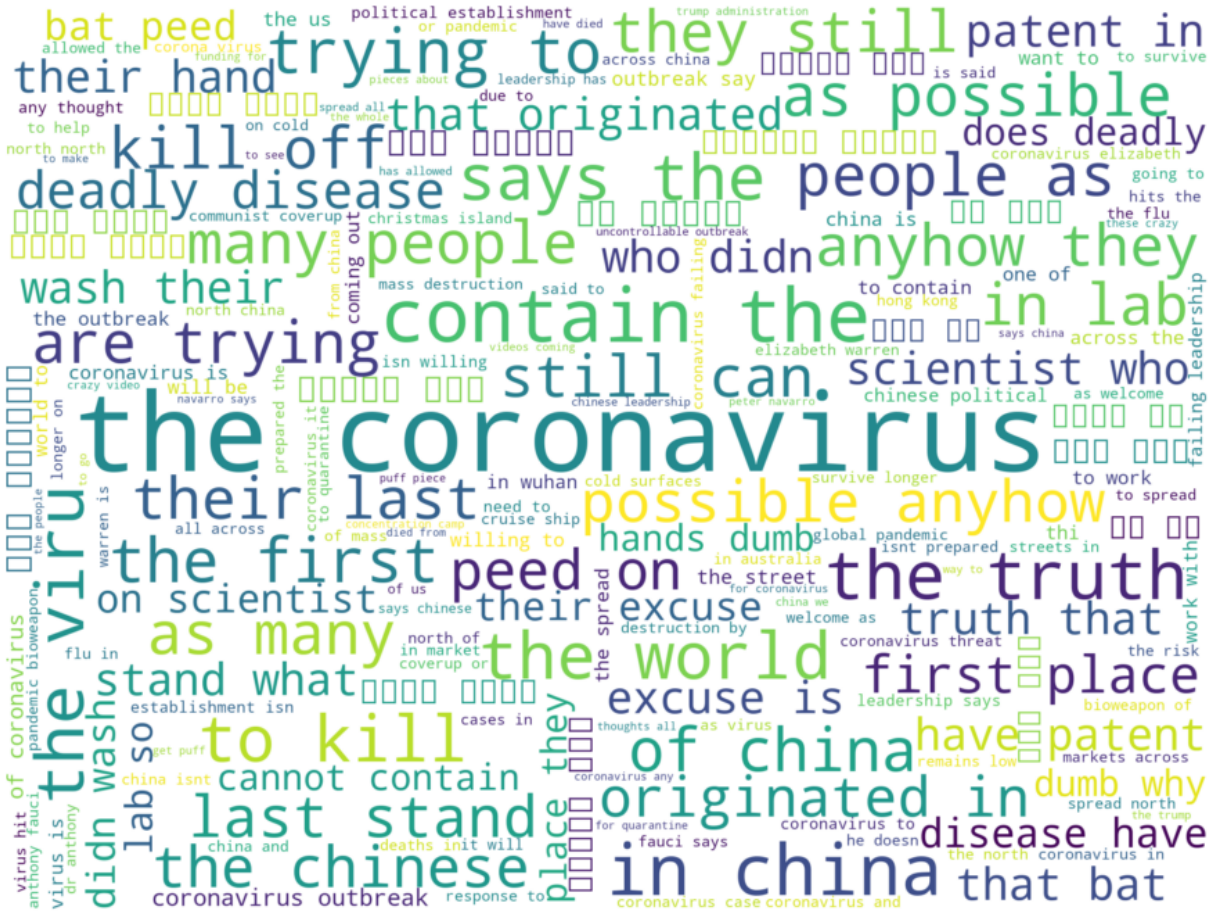


Figure 4.14: Word map of commonly occurring phrases and text that co-occur with hashtags from period 1, cluster 6, which is composed of U.S. politics related hashtags. There is a wide range of verbiage employed in the cluster, with much of it focusing on the COVID-19 pandemics origins in China and its subsequent spread.

Generally, no phrase or word is particularly dominant in this cluster except for 'the coronavirus' itself. Much of the text mentions the origin of the pandemic in China as well as the global spread of the virus. In this first period U.S. politics cluster, much of the verbiage centers around the origin of the virus and its possible effects. So, much of the discussion from politically-connected hashtags during the first period focuses on China's role in the origin and spread of the COVID-19 pandemic.

Turning to the third period U.S. politics focused cluster, there are similar usage patterns among the hashtags. The following table, Table 4.7, displays the salient hashtags from the third period U.S. politics cluster.

Period 3, cluster 5: U.S. Politics					
Most Used Hashtags	Number of Uses	Hashtag with Highest Original User Ratio	User Ratio	Hashtags with Lowest User Ratio	User Ratio
Trump	189109	Satanism	0.988	hillaryemails	0.030
FakeNews	74076	DeutscheBank	0.987	PoliticalViews	0.037
MAGA	70999	Morons	0.986	HAction	0.040
FoxNews	67531	Socialists	0.981	drudge	0.051
WWG1WGA	49348	ShutItDown	0.978	slate	0.064
KAG	48339	AlexJones	0.977	newsabq	0.065
Trump2020	46790	2ndAmendment	0.971	abqfm	0.067
OneVoice1	43457	hypocrisy	0.969	rockoftalk	0.076
QAnon	41446	DrainingTheSwamp	0.960	NewsVideo	0.079
CoronavirusUSA	34819	Bullshit	0.960	bluelivesmatter	0.083

Table 4.7: Important hashtags from period 3 cluster 5 which has U.S. Politics related hashtags. As with other U.S. politics clusters, this cluster is a mix of hashtags from political news sources, political personalities, and politically-motivated conspiracy theories. Relative to the first period's U.S. Politics cluster, there is an increase in the use of more politically inflammatory hashtags.

As with the first period's U.S. politics cluster, many of the hashtags surround political commentary, politically-motivated conspiracy theories, and high profile politicians. Unlike the first period, however, the hashtags with the lowest user scores have larger users bases (i.e. 1 unique user versus 7 for the lowest scoring hashtag). There is also an increase in the number of hashtags being used, 343 in period one versus 610 in period three. To get a better sense of the differences between the two periods, the verbiage of the text co-occurring with the hashtags was then analyzed. The following figure, Figure 4.15, displays a word map of the commonly used phrases and words from the cluster

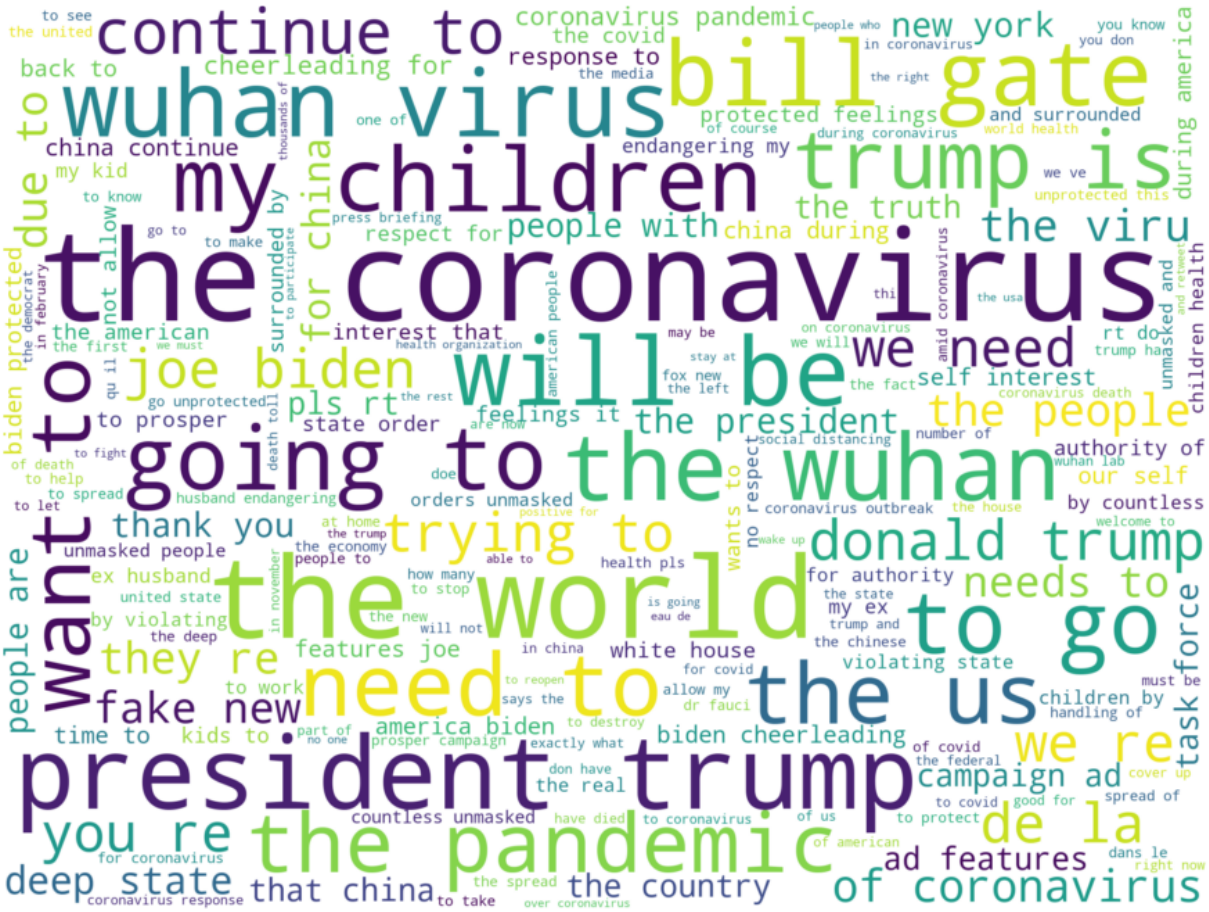


Figure 4.15: Word map of the commonly used phrases and words from period 3 cluster 5, which consists of U.S. politics related hashtags. Much of the verbiage focuses on personalities, like President Donald Trump or Bill Gates who has been erroneously linked to the COVID-19 outbreak by conspiracy theories.

In this time period phrases surrounding personalities become much more used, most especially President Donald Trump. Other personalities like Bill Gates and Joe Biden are also frequently mentioned in the political hashtag tweets. So, from the first period to the third period, much of the verbiage shifts from a focus on the virus' origins in China and its spread, to personalities who are often political in nature or who have been linked to the coronavirus, even if erroneously, such as Bill Gates. So, overall there is a shift in the focus of the verbiage paired with the U.S. politics hashtags over the course of the pandemic. This shift generally goes from an external focus to an internal focus relative to the United States.

Finally, to get a more complete sense of the differences between the period 1 and period 3 clusters, we directly compare the membership of these two clusters. The user base overlap between the two time periods is small; only 8% of the union of the two time periods' users are present in both time periods. These users are generally high profile users like politicians or news sources. There is also a 31.8% overlap in the hashtags used between the two time periods. To get a sense of the differences in membership between these clusters, I analyzed the most important hashtags that exist in only one of the two clusters. It should be noted that many of the hashtags that exist in only one cluster do exist in the other time period that their cluster does not belong

to, but not within the U.S. politics cluster of that time period. The following table, Table 4.8, displays a side-by-side comparison of the hashtags between the two clusters.

Period 1 Only Hashtags	Number of Uses	Period 3 Only Hashtags	Number of Uses	Period 1 Only Hashtags	User Ratio	Period 3 Only Hashtags	User Ratio
coronavirusaustralia	35561	smartnews	28748	openborders	1.000	morons	0.986
newyork	7131	5g	27165	newyork	0.999	socialists	0.981
racism	5016	america	18568	thegreatawakeing	0.994	shutitdown	0.978
censorship	2426	oann	16456	bias	0.946	2ndamendment	0.971
thegreatawakeing	2017	pressbriefing	11899	virginia	0.943	hypocrisy	0.969
coronovirus	1829	new	8650	trumpbudget	0.939	justasking	0.959
trumpbudget	1689	nyt	8276	chaos	0.939	nyt	0.959
democracy	1613	deepstate	7982	confirms	0.926	senatorforsale	0.958
zerohedge	1574	senatorforsale	7389	lnpfail	0.923	justsaying	0.956
iran	1546	americans	7113	earthquakes	0.922	antivaxx	0.955

Table 4.8: Comparison of the important hashtags that either in the U.S. politics cluster in period one or period three, but not both. Generally, there is an increase in conspiracy-related hashtag and inflammatory hashtag usage from period 1 to period 3.

There are some distinct differences in those hashtags only used in one of the clusters and not in the other. First, the salient period one only hashtags feature Australian-related hashtags like '#coronavirusaustralia' and '#lnpfail' which are not in period 3. Also, there is a rise in the use of conspiracy-related hashtags in the third period only hashtags, such as '#5g', '#deepstate', '#antivaxx' and more inflammatory hashtags like '#morons' or '#senatorforsale'. So, there is not only a regional shift in terms of the difference in the U.S. politics hashtags between periods one and three, but also one toward more polarizing and contentious hashtags over time as well.

Overall, the cluster of U.S. politics-related hashtags differs over the course of the pandemic. The use of some hashtags (approximately a third) remains the same, but the verbiage associated with those hashtags changes from a spread of the disease and Chinese focus to a personality and conspiracy-theory focus. Additionally, the hashtags used in only one time period also show some distinct differences between the time periods. So, while an easily defined topical discussion characterized by the hashtags being used can be persistent over the course of the pandemic, the nature of that topical discussion cluster changes. It is also worth noting that known conspiracy theory related hashtags are always present in the U.S. politics cluster, which demonstrates an strong connection between the two over the course of the COVID-19 pandemic.

4.4 Discussion

There are several findings from this study and results to inspire future research. First, through a scalable technique, like MVMC, it is possible to extend multi-view clustering to a task like clustering hashtags in large-scale social media data. Large scale social media data often requires clustering of tens or even hundreds of thousands of objects and the ability to handle partially incomplete data. The MVMC procedure can successfully deal with both of these conditions in the data and produce meaningful clusters. Use of the MVMC technique also found that certain views, in their current form of feature representation, like URLs which co-occur with hashtags

in tweets were not useful in finding a cluster structure in the hashtags. Additionally, the MVMC technique converged in every case and converged in less than eight iterations which, based on previous empirical findings, support the belief that there is a cluster structure present in the hashtags. That is to say that usage of hashtags is not uniformly at random and that groups of hashtags tend to be used in similar fashion for similar purposes which supports the idea of using hashtag clusters to understand conversation topics. Also the use of multi-view clustering on hashtag can ameliorate problems with previous attempts to cluster hashtags based on just one view. For example, incorporating text and shared users can overcome the observed phenomenon where two hashtags are very related in usage, and should be clustered, but are never used by the same users. Thus, through a technique like MVMC it is possible to incorporate all of the previous research on clustering hashtags, that have used co-occurring text or users, into one cohesive model and clustering.

Second, hashtag usage patterns during the COVID-19 pandemic displayed dynamic behavior at both the individual and cluster levels. While the data collected is certainly an incomplete picture of the discussions happening on twitter due to API restrictions and the terms used to create the data, it is still large enough to offer some insights. From the early days of public awareness about the pandemic in February of 2020, there was an increase in the number of unique hashtags being used and the the number of unique users participating in the COVID-19 discussion on twitter. The usage patterns of hashtags, however, varied over the course of the pandemic with there being an initially high rate of hashtag usage that drops when the number of unique users increase, and then increases again as the number of unique users levels of in late March/ early April. This suggests that when a major exogenous shock happens to social media users, like a pandemic, there will be an initial phase of interaction without hashtags, and then a move to start re-using hashtags, likely as a tool to aid in finding and participating in discussions.

At the cluster level, the data showed there were three main periods of clusters of hashtags present in the data. The daily hashtag clusterings could themselves be clustered into three distinct time periods of clusterings based solely on the pairwise similarity between the daily clusterings. In general, the cluster structure of hashtags went from a large number of small clusters to fewer, larger clusters and then back to smaller more numerous clusters. This macro temporal pattern in the cluster structure mirrors those findings from the use of individual hashtags and supports the conclusions that there was a surge in COVID-19 twitter discussion which produced an intermediary period of hashtag usage which then settled into a new pattern of hashtag usage different from what was observed prior to the user surge.

Using the knowledge that the daily hashtag clustering breaks into three periods, I then created an ensemble clustering for each of these periods. This ensemble clustering allows for a clustering analysis of a prototypical clustering for the entire time period. The results of the analyses of these ensembled clusterings produced some interesting insights into the nature of some of the topical discussions happening during the pandemic. Firstly, there are some topics which have been persistent over the course of the pandemic, like commerce, the economy, U.S. politics, and news. Other topical groups like online education or negative-sentiment discussion about the Chinese government are more transitory over the course of the epidemic. From these clusters it was also observed that some topical groups are intending to direct COVID-19 discussion to other topics like the Syrian Civil War or protests in Hong Kong. So, it would seem from the nature of the clusters present that there is the presence and use of hashtags that are meant to use the COVID-

19 pandemic to draw attention to other causes or ideas. So, hashtags can not only be a means helping users to find and participate in discussions but also as means of shaping user engagement and the discussions themselves.

From the results presented in this chapter there are several avenues for future research. First, this study focused on the use of hashtags in order to understand topical discussions taking place, which naturally discounts users who do not use hashtags. As was seen in the data section, there is a sizeable amount of the population of users that are posting content related to the COVID-19 discussions that do not use hashtags. So, a future area of research would be to look more broadly at the concepts that users are employing in their tweets. So, instead of just clustering on hashtags, one could look at clustering on hashtags and topical labels from something like the tweet text. Second, for the multi-view clustering of large scale clustering of hashtags in social media data there is a need for future research on the appropriate views and how to feature engineer those views to be useful. The URLs view of the data ended up being unhelpful for the found clusters, and this seems to be due in large part to the fact that there was very little overlap on exact URLs, due to things like URLs including query terms or from the inability to resolve shortened URLs. So, there were situations in which essentially the same story or piece of news was used with two different hashtags in two different tweets, but because the URLs were not exactly the same, those hashtags were not recognized as being similar by the method. So, a means of processing the URLs to do something like just using the top level domains should be tried in future work. Additionally, previous research on misinformation during the COVID-19 pandemic has demonstrated that it can spread quickly by mechanisms like retweeting. It would be of value to create a tweet type view to characterize what type of tweets are being used with certain hashtags. Such a view may help with distinguishing between clusters of hashtags used for misinformation versus those used for more legitimate information. Second there is also potential for future research in better cleaning and representing real-world data for multi-view clustering. For example, the tweets used in this study were not filtered by language. Performing a filtering step like only using English-language tweets could lead to more nuanced and meaningful clusters. Additionally, I adopted a simple and scalable graph learning procedure to form the view graphs for multi-view clustering of this data due to scalability issues with many of the more sophisticated graph learning procedures. Thus, an important avenue for future research is to find a graph learning procedure for MVMC that can better fit the intrinsic structure of the data, but that is also scalable to hundreds of thousands of entities. Finally, the data used in this study was only a sample of the twitter data pertaining to COVID-19. It would be interesting to see if different COVID-19 twitter data yield the same results and if there are differences between different social media platforms that also employ hashtags.

Chapter 5

Case Study: Characterizing Communities of Malware Use

In this chapter, I use the previously described multi-view clustering techniques to analyze malware samples. Some of the most destructive use of malware has been by threat actor groups [120]. These threat actor groups can be considered a special type of social community where the community often has specialized knowledge and engages in particular behaviors. In this chapter multi-view clustering techniques are used to analyze the community structures of threat actor groups by their malware usage. I show that the used of the Hybridized technique of Cross-View Influence Clustering (CVIC) provides the best ability to identify communities that mirror the threat actor groups. I also demonstrate that the use of hybrid, late, and intermediate multi-view clustering techniques outperform earl integration multi-view clustering.

5.1 Background

Malware continues to be one of the most prolific and destructive threats to cyber systems. Malware is also often a tool of choice for threat actor groups that seek to compromise various cyber systems [120]. The volume of malware produced continues to accelerate resulting in thousands of new samples of malware being discovered everyday. The vast majority of malware samples are actually variants of existing malware, produced by transforming or obfuscating an existing sample in such a way that it can evade detection by security products and other defenses. As such, many malware samples could be characterized as being a malware community or family with distinct relations between samples. Furthermore, many malicious actors employ different types of malware (i.e. Worms, Trojans, Viruses, etc.) that have certain distinct patterns in their features that can aid in identifying which actor a malware sample may have come from [29]. So, while there are millions of malware samples, they tend to congregate into families which may only be used by a certain threat actors. Consequently, the categorizing of related malware into possible threat actor groups and understanding relationships between the malware used by different threat actors can aid malware analysts and security operations in prioritizing responses and defenses for new malware-based attacks.

While there are a number of existing techniques for analyzing malware [149], one approach

that has not been explored significantly involves analyzing features from malware samples from a *multi-view* perspective. That is to say there are many types of features, modes or views, from static to dynamic or hybrid, that could be used to identify different types malware and possibly characterize them into threat actor groups. Most work on using these various features of the malware to date use an *early integration* approach with these features. Furthermore, most research on identifying and characterizing malware samples focuses around placing malware samples into malware families; few works to date try to identify the threat actor(s) that are using and developing the malware [149]. In this work, I approach the task of characterizing samples of malware into possible threat actor groups, without any prior knowledge of possible threat actor groups, as an intermediate or late integration multi-view, clustering problem. I demonstrate both the effectiveness of intermediate, hybrid, and late integration multi-view approaches to clustering samples of malware into threat actor groups and multi-view clustering techniques using several hundred heterogeneous malware samples from three different threat actors: Dukes, APT-1, and Deep Panda. The main contributions of this chapter are summarized as follows:

- The performance of the different multi-mview clustering algorithms is demonstrated using heterogeneous malware samples from three different malware threat actors. The hybrid integration technique of CVIC produces the best communities from malware samples in terms of the samples' threat actor groups.
- Clustering malware samples using intermediate, hybrid, or late integration multi-view approaches, rather than early integration, consistently shows better results in malware threat actor characterization.

The chapter is organized as follows: In the next section I review some of the work related to characterizing malware families and multi-view clustering. In Section 3, I outline the methods used to extract features from malware samples and to create graphs the clusterings from the features. In Section 4 I present the results of testing the algorithms on malware data. Finally, the last section concludes with some reflections on the results and presents avenues for future research in multi-view clustering of malware.

5.1.1 Related Work on Unsupervised Learning and Malware

The characterization and identifications of malware samples remains an active area of research. Many of the current methods today break down into whether they employ supervised or unsupervised machine learning and what types of features they use [149]. In terms of the features being used to characterize or identify malware, most features come from either static or dynamic analyses of the malware [149]. Dynamic features tend to focus on malware behavioral patterns, whereas static features tend to be aspects of the malware sample itself [149]. One of the most promising set of static features from a malware classification standpoint were those proposed by Berlin and Saxe in [111]. These features have been shown to give state of art performance in supervised learning of malware samples [111] as well as being useful for characterizing malware samples within a threat actor group in an unsupervised learning paradigm [36]. Despite the prevalence and variety of malware features available, most approaches in use today use an early integration approach with the different views of the data. That is to say, current methods take all of the views of features for identifying the malware and concatenate (typically with

some kind of normalization or feature engineering across views) all views of the features into one feature vector and then feed those vectors into standard machine learning algorithms. So, while characterizing malware samples remains an important problem for things like cybersecurity countermeasure or attribution, nearly all of the clustering of malware samples is done as a standard clustering problem.

Outside of early integration approaches there have only been couple of techniques that have been employed to cluster cybersecurity threats. Most of these techniques employ some kind of cluster ensembling [18]. One recent paper used modified versions cluster ensembling algorithms originally proposed by Strehl and Ghosh in [119] to combine some static and dynamic malware features for better malware clustering [62]. Another technique used features from Phishing websites along with the malware present within the Phishing website to characterize Phishing websites into communities and even assign attribution [159]. It should be noted however, that while their proposed technique included cluster ensembling, it is actually an early integration technique for multi-view clustering. The authors concatenated the features from the Phishing websites and malware into one set of features, and then used cluster ensembling. Overall, there has not been much employment of multi-view clustering from late or intermediate integration paradigms for analyzing malware.

In summation, the characterizations of malware samples whether into families or into more complex groups like threat actors remains an area of active research, and one that uses many different types of data to characterize and identify malware. There has been little work analyzing malware data from a multi-view data perspective using other than early integration techniques or exploiting both clusterings and graphs within the same multi-view clustering algorithm.

5.2 Method

The overall method of characterizing malware samples consists of three, main steps: 1) Extract the different features of the malware using the static analysis from Berlin and Saxe [111]. 2) Find the cluster assignments and nearest-neighbor graphs for each of the modes of the data. 3) Use a cross-diffusion, influence process to update the cluster assignments for each sample, and cluster the resultant bipartite graph for the final cluster assignments (i.e. CVIC). The first steps are existing means of dealing with malware data and graph learning respectively, so I summarize them in the Background section for completeness. The third step of Cross-View Influence Clustering is summarized in section subsection B. The following figure, Figure 5.1, graphically summarizes the entire method for characterizing communities in the malware samples.

The use of the CVIC technique from Chapter 2 is empirically driven. First, CVIC, along with IPMMC, were the only two algorithms found to be robust across all of the social-based data scenarios. And, since clustering malware samples does not match of the social-based data scenarios seen before, I have opted for one of the more robust empirically robust techniques. Second, as will be seen in the results, section, CVIC produced the best performance across all of the techniques.

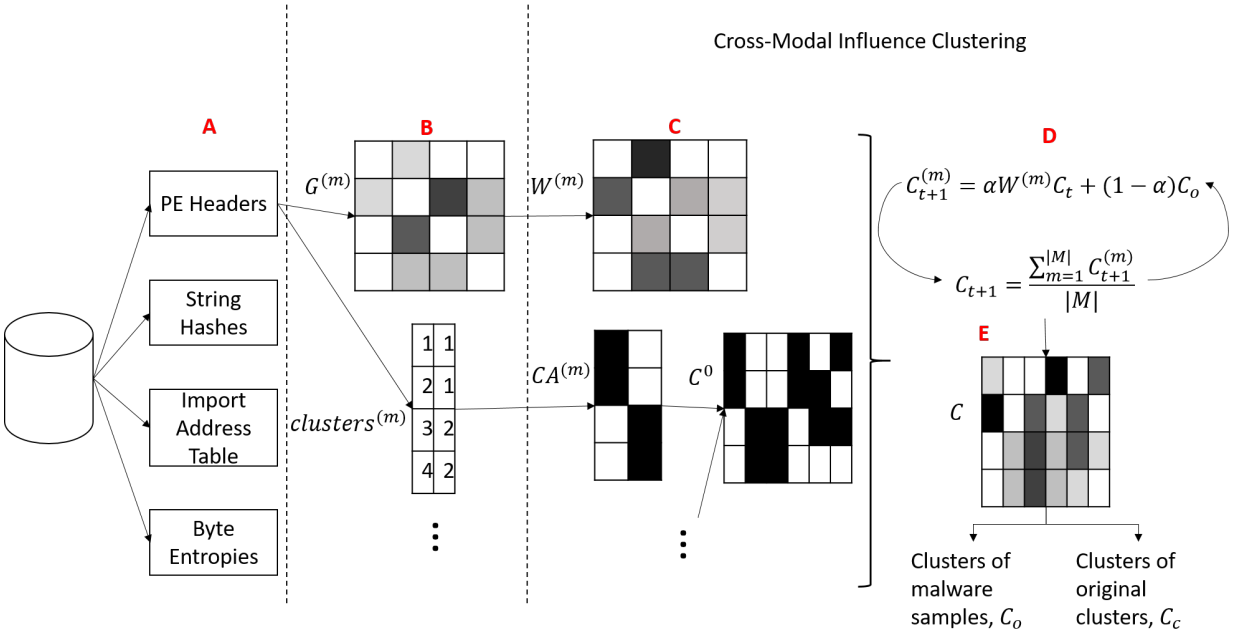


Figure 5.1: Graphical Overview of the Methodology for clustering malware samples into threat actor groups. In step **A** I extract four different sets of features from the malware samples using the methodology originated by Berlin and Saxe [111]. Following the feature extraction, In **B** I use a *Mod-kNN* procedure to find both cluster assignments and nearest neighbor networks, G, for each of the sets of features. Then, in **C-E** I use hybrid integration methodology, *CVIC*, which is based on concepts from network diffusion and social influence to get the final cluster labels for the malware samples.

5.2.1 Background

Since the hybrid integration approach relies on both clustering assignments and a graph representation for each view of the data, I now detail how view clusterings and graphs are obtained in this section. For the sake of completeness, I also provide information on how the Berlin and Saxe [111] features are extracted from malware samples.

Obtaining Features from Malware Samples

The features from Berlin and Saxe come from a deep neural network-based extraction method that uses four static feature domains. In previous works these features have been show to achieve high accuracy malware classification results [111], as well as provide insight into malware communities present in a threat actor group through clustering [36]. The four categories of features are Byte Entropies, Import Address Table, String Hashes, and Portable Executable (P.E.) headers. The Byte Entropies are produced by concatenating the row values of a two-dimensional histogram constructed from byte-entropy value pairs of all bytes in a given 1024-byte sliding window. New entropy values and byte-entropy pairs are computed for each window, which traverses the file using a step size of 256 bytes. The Import Address Table Features are generated by hashing library and library function names to the range $[0, 255)$, and measuring the frequency counts of each resulting hash value. The String Hashes features are produced by by hashing all printable strings of length 6 or more (in the ASCII printable range) to the range $[0, 16)$, pair-

ing each string's hash with the log of its length, constructing a two-dimensional feature vector mapping those hash value / log-length pairs, and then concatenating the rows of the histogram. Finally, the P.E. Headers is produced by extracting numerical features from a file's Portable Executable packaging and aggregating those features into a 256-length array. In total, four different sets of features that are all 256 dimensional vectors of non-negative, integer values are extracted in the first step.

Obtaining k-Nearest Neighbor Graphs and Cluster Assignments for Each Mode

Since we are interested in the clusters present in the data, we want graphs (and clusters) that highlight any cluster structure present in the data. One way of doing this is to explicitly learn graphs that emphasize in clusters in the data. So, for the first step of finding clusters and nearest neighbor graphs within each view of the data, I employ the modularity k-Nearest Neighbor graph (mod-kNN) procedure [110], [26]. At a a high level, mod-KNN method iterates through various possible numbers of neighbors for each vertex, k, and selects that k which produces the most *modular* graph, relative to a null-model, random graph produced on the same set of vertices. It then outputs this graph and the clustering assignment for each of the vertices that maximizes the modularity. Modularity in this case is the network modularity as described in [94]:

$$Modularity(G) = \frac{1}{2n}\sum_{ij}[A_{ij} - \frac{deg(i) \times deg(j)}{2n}\delta(c_i, c_j)] \tag{5.1}$$

where v is the number of vertices in the graph, and c are the cluster assignments of the vertices. Since it is known that random graphs can give rise to modular structures, I subtract the modularity obtained from a random graph with the same number of vertices and edges from the modularity value from clustering the derived graph. I further incorporate the changes to the original algorithm from other works, namely to use asymmetric k-NN graphs for each value of k (wherein an edge exists between two nodes, u and v, if either $u \in kNN(v)$ or $v \in kNN(u)$) and the Louvain method for clustering the kNNs [25], [26], [15]. Finally, I note that the Louvain algorithm, while very fast for clustering a graph, is also a greedy, stochastic algorithm that can lead to different clustering outcomes for a graph depending on the initialization of the algorithm. So, I also run the Louvain algorithm a number of iterations on each asymmetric kNN graph and evaluate that value of k by the average modularity of each of the iterations of Louvain clustering [15]. This is done to provide for greater stability in both the selection of the kNN graph and the resulting cluster assignments by the procedure. The psuedo-code of my implementation of network construction by mod-kNN is detailed in algorithm 12.

Algorithm 12 Modularity k-Nearest Neighbor Graph

input: Distance or Affinity Metric, s, number of iterations to run Louvain method, l_{iter}
output: Optimal k-Nearest Neighbor Graph, G^*, and sub group assignments $C(G^*)$.
for $i = 1 : \lfloor log_2(n) \rfloor$ **do**
  $k \leftarrow 2^i$
  $G_k \leftarrow kNN(s, k)$
  $G_k \leftarrow maximum(G_k, G_k^T)$
  $G_k^r \leftarrow randomize(G_k)$
  for $l = 1 : l_{iter}$ **do**
    $C_l(G_k) \leftarrow Louvain(G_k)$
    $C_l(G_k^r) \leftarrow Louvain(G_k^r)$
  end for
  $Modularity_k \leftarrow \frac{1}{c_{iter}} * \sum(Modularity(C(G_k)) - \frac{1}{c_{iter}} * \sum(Modularity(C(G_k^r)))$
end for
$k^* \leftarrow \text{argmax}_k \, Modularity_k$
$G^* \leftarrow G_{k^*}$
$C(G^*) \leftarrow \text{argmax}_l \, C_l(G^*)$
return $C(G^*)$, G^*

In Algorithm 12 an asymmetric kNN graph is created by doing a point-wise maximization with the transpose of the raw kNN graph which is obtained through any subroutine that finds the k-Nearest Neighbors of each data point given a means of calculating similarity or difference between the data points, $kNN(*, *)$. In this way, the graph becomes undirected which is critical for some algorithms that I investigate in the results section. Additionally, while I have represented the iterations of the different Louvain clusterings of the kNN graph within a for loop, this operation can easily be parallelized for performance. Finally, I evaluate the modularity of each possible value of k by using the average difference in modularity, but select the final cluster assignments for the final k, $k*$, by selecting these assignments from the Louvain iterations that gave rise to the best difference in modularity.

5.2.2 Cross-View Influence Clustering

In this section, I will outline in more detail the CVIC algorithm originally proposed in Chapter 2. Given a graph representation of each view of the data along with the optimal cluster assignments for each mode, I use a cross-diffusion process [134] in a social influence model [51] to get the final cluster assignments for each sample. Cross-View Influence Clustering (CVIC), iterates through two main steps: 1) update the cluster assignments for each view by diffusion in a social influence model with the cluster association matrix and view graphs. 2) Combine the updated cluster assignments from each view into a new cluster association matrix. Once the iterations are complete, the result is a new cluster association matrix that relates each entity to each of the clusters. This matrix naturally forms a bipartite graph of all of the entities by the cluster labels, which can be clustered by any bipartite graph clustering technique.

For the first step in CVIC procedure, I use a diffusion model inspired by social influence

models. Social influence models are often used to simulate the propagation and formation of belief clusters within social groups [49], [51]. As I am interested finding clusters of malware samples, I use view clusterings as 'beliefs' and the modal graphs as the means by which samples may influence each others' beliefs. In particular, I use the Friedkin social influence model described in [51] due to its demonstrated utility and stability in terms of outcomes. Friedkin's social influence model is given by:

$$C^v_{t+1} = \alpha W^{viewv} C_t + (1 - \alpha) C_0 \tag{5.2}$$

where C is an object by cluster association matrix of size number of malware samples by total number of clusters across all modes. A cluster association matrix is an object by cluster matrix whose entries represent how strongly a particular object is related to a particular cluster. W^v is the row-normalized modal graph from view v and α is a influence parameter that balances the strength of influence over belief in cluster labels coming from neighbors in a view versus the initial beliefs in clustering labels. It should be noted that this influence model closely resembles those used in metric learning, especially for image retrieval [43].

After having performed the influence step for each mode, I then need to combine the view clustering association matrices, C^v_{t+1}, to create singular cluster association matrix C_{t+1}. To do so, I adopted the simple averaging technique used in cross diffusion [134] and bipartite graph partitioning for ensembling [47]:

$$C_{t+1} = \frac{\sum_{v=1}^{m} C^v_{t+1}}{m} \tag{5.3}$$

where M is the collection of modes and $|M|$ is the number of modes. The cross diffusion process diffuses a graph in each of the modes separately and then uses that newly diffused graph in all of the other modes than the one which produced it for another diffusion step [43]. In this way information from mode is spread across all of the modes, and the complementarity of information between the modes is better used then trying to diffuse information in each mode separately. The two steps of influence and aggregation are repeated for a specified number of steps and the resulting cluster association matrix, C_{final} is then clustered as a bipartite graph as has been done in other cluster ensembling techniques [47], [64]. The pseudo-code of the CVIC procedure is detailed below, in Algorithm 13

Algorithm 13 Cross-View Influence Clustering

input: Graph for each view G^v, cluster assignment for each view $clusters^v$, Social influence effect, α, and number of iterations, $iterations$
output: Cluster assignments for objects, C_o, and for the original clusters, C_c.
for $v = 1 : m$ **do**
  $C^v \leftarrow cluster\_association(clusters^v)$
  $W^v \leftarrow row\_normalize(C^{vT})$
end for
$C^0 \leftarrow concatenate(C^v \forall m \in M)$
$C \leftarrow C^0$
for $t = 1 : iterations$ **do**
  for $v = 1 : m$ **do**
    $C^v \leftarrow \alpha \times W^v C + (1 - \alpha) \times C^0$
  end for
  $C \leftarrow \sum_{m=1}^{|M|} \frac{C^v}{|M|}$
end for
$C_o, C_c \leftarrow bicluster(C)$
return C_o, C_c

As a first step in the algorithm I create cluster association matrices C^v for each view of the data v for all of the total number of modes m. It is row-stochastic by construction ($\sum_i C_{ij} = 1, \forall j$), and in the case of crisp clustering, it will also be a binary matrix with only one entry per row being equal to 1 and all others being equal to 0. I also transform each graph into a transition matrix by making it row-stochastic ($\sum_i W_{ij} = 1, \forall j$). From there, all of the view clustering association matrices are concatenated into one cluster association matrix C, which is of size number of entities by total number of clusters across all modes. This cluster association matrix then undergoes an iterative two-step process that results in the final cluster association matrix. In the first step, I use a social influence model to update the cluster associations within each view of each object by a convex combination of its neighbor's cluster associations and its original cluster associations, $C_i^v = \alpha \times \sum_{j \in N_m(i)} W_{ij}^v C_j + (1 - \alpha) \times C_i^0$, where i is a particular object and j are the neighbors of that particular object in a particular view, v. Following this step, I then combine all of the social influence updates into one cluster association matrix. This cluster association matrix then becomes the input to the next iteration of the social influence model. After all iterations are finished, I then use a biclustering algorithm such as Bi-Louvain [155], to cluster the final cluster association matrix. In this way, I follow the same procedure as employed in cluster ensembling, where the object by cluster association matrix is clustered to produce a final, more robust clustering [64], [18]. The outcome is then not only the final cluster assignments for each objects but cluster assignments for the original clusters across all modes.

5.3 Results

In this section, I analyze the community detection properties of CVIC against several other multi-view clustering techniques. I also look at the multi-view nature of the malware data and the

robustness of the CVIC algorithm to changes in the user-set hyper-parameters. In order to empirically evaluate the various algorithms I used heterogeneous malware samples from three different threat actors. The first malware threat actor is Deep Panda, which is a nation-state backed actor who targets a wide range of industries, including government, defense, financial and telecommunications [5]. In this data all of the Deep Panda malware samples belong to the Sakula malware family which are variations of remote access Trojan (RAT) tools [40]. The second set of samples comes from the APT-1 threat actor which uses various types of malware to attack many different systems in government and industry [29]. The third set of samples comes from the Dukes threat actor which uses various types of malware to attack various targets in support of espionage and security policy decision making for a nation-state [45]. In total, empirical testing was done with 2,185 different malware samples with 288 from the Deep Panda group (Sakula malware family), 973 coming from the APT-1 group, and 924 coming from the Dukes group. Within the the Dukes samples, 83 of the samples have the actual malware family that the sample belongs to (e.g. CosmicDukes, PinchDukes, etc.). It should be noted that the types of malware present (i.e. Trojan, Worm, Virus, etc.) are not necessarily homogeneous within any given threat actor's samples. APT-1 and Dukes contain a greater variety of types of malware than do the Deep Panda threat actor which just has samples from the Sakula family of malware .

5.3.1 Community Detection Results

In this section I analyze how well various means of clustering the malware data can be used to characterize malware samples into threat actor groups. In order to get as thorough analysis as possible of detecting threat actor groups from heterogeneous samples of malware, I have have chosen to use several different multi-view clustering techniques that were used in Chapter 2. The following table, Table 5.1, summarizes the techniques and the multi-view clustering paradigms that the techniques come from.

Intermediate Integration	Hybrid Integration	Late Integration
CNDC_I	DISC_A*	GP-MGLA
SFI*	DISC_M*	CSPA+
SPSL*	DISC_R*	BGPA+
CG*	DICL_A	LWBG
ResK*	DICL_M	MCLA+
IPMMC	DICL_AG	LWMC
NF-CCE*	DIMC_M	DREC*
ETL-MSC*	MSIM_C	LPMMC
	CVIC	
	CNDC_C1	
	CNDC_C2	

Table 5.1: Summary of the multi-view clustering methods used in evaluating the malware samples. Full descriptions of the methods are available in Chapter 2.

For all of the multi-view techniques the cluster assignments and graphs for each view were

determined using the mod-kNN procedure, Algorithm 12. This ensured that each technique was using the same starting basis for a fair comparison and reasonable versions of both the view graphs and view clusterings. For these experiments, the mod-kNN similarity metric used was the Bray-Curtis dissimilarity [19], given by:

$$s_{ij} = \frac{|X_i - X_j|_1}{|X_i + X_j|_1} \tag{5.4}$$

where i and j are two different objects (malware samples) and X_i and X_j are their respective feature vectors. I also tested Euclidean distance, cosine distance, and the Jaccard Index, but found Bray-Curtis to give the best results for the mod-kNN procedure with the Berlin and Saxe malware features.

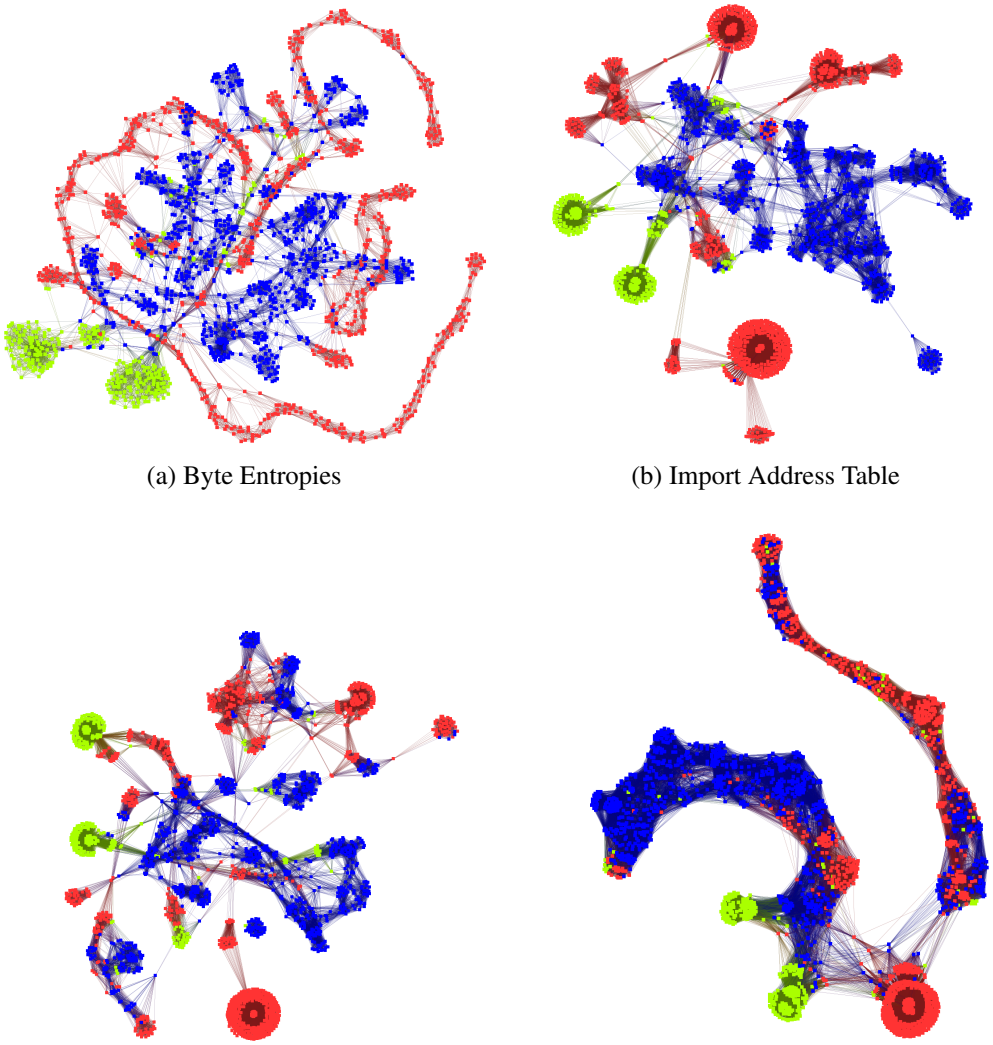
5.3.2 Graphs Learned From Malware Features

The graphs formed by the mod-kNN procedure from the four different types of malware features have distinctly different topologies. In order to characterize the topologies the following table, Table 5.2, summarizes the graph measures for each of the views of the malware data.

Feature	Density	Number of Components	Avg. Clustering Coefficient	Isolates	Average Degree	Degree Assortiativity
Byte Entropies	0.001	1	0.68	0	21.3	0.14
Import Address Table	0.022	1	0.79	0	47.8	-0.17
String Hashes	0.042	1	0.69	0	92.2	-0.084
PE Headers	0.021	2	0.79	0	46.6	-0.122

Table 5.2: Summary of the topological properties of the learned graphs from each of the views of the malware. In each case, each graph has relatively string community structure, but also wide differences in edge statistics. This would indicate that the patterns are different between the views of features.

Overall, all of the views have strong clustering coefficients, which correlates with a strong cluster structure being present in the graphs [94]. From the empirical findings in chapter two, this high average clustering coefficient also indicates that these graphs should also work well with a multi-view clustering technique. The graphs are, however, very different in topology with respect to the degree assortiativity, average degree, and density, despite all being formed by the same process. These differences in edges statistics would indicate that there are different patterns of structures present within each of the views. So, combining all of these features together could obfuscate these view patterns which could be useful for detecting clusters in the samples. So, this would suggest that the use of multi-view clustering should be successful on these malware samples. To get a better idea of the differences in the graphs between the different views of the malware, the following figure, Figure 5.2, depicts the actual view graphs.



(a) Byte Entropies

(b) Import Address Table

(c) PE Headers

(d) String Hashes

Figure 5.2: Graphs of the four different views of the malware samples. Vertices in red are the Dukes malware samples, blue the APT-1 malware samples, and green the Deep Panda malware samples. In all views, the Deep Panda samples display two distinct communities.

The different view graphs display distinctly different topologies. However, Across the different topologies, some distinct patterns emerge from the images. For one, the Deep Panda malware samples, which are all of the Sakula virus, have two distinct clusters in ever view. This suggests that even though the Deep Panda samples are all one homogeneous family of malware, that this malware has some distinct differences within the family. This result is largely consistent with other analysis of the communities in Sakula virus family of malware [36]. Also, in all views except the Byte Entropies, the Dukes malware also displays a distinct, large community. Thus, the different malware manifest community structure differently across the different views.

5.3.3 Clustering Results

In this section I analyze the ability of the various techniques to cluster the malware samples into threat actors. Since all current research on malware data represents the samples' features as being uni-modal (i.e. takes an early integration approach), I have also chosen to test various common standard clustering techniques after concatenating all of the modes of features into one mode. In particular, I create a uni-modal feature representation of each malware sample by simply concatenating the four Berlin and Saxe feature vectors into one feature vector.

To evaluate the success of the different methods I have chosen to evaluate the clustering outputs by the Adjusted Mutual Information (AMI) [132] and Adjusted Rand Index (ARI) [69] of each method's final clustering with the ground truth threat actor labels of each malware sample. Both of these measures give an indication of how well each clustering matches the ground truth clustering, with higher values indicating better performance. Also, I measured each method's Silhouette score, which is a measure of both how compact and how separate each cluster is, given the original feature vectors [109]. The Silhouette score does not rely on ground truth knowledge of the cluster assignments and is thus regularly used as a means of assessing the goodness of a clustering method in the absence of ground truth labels. Finally, I ran each method 20 times and report the average of all of their scores. The following table, Table 5.3 summarizes the results.

Clustering Technique	ARI	AMI	Silhouette
GP-MGLA	0.2	0.4	-0.51
CSPA+	0.23	0.37	-0.68
BGPA+	0.22	0.41	-0.53
LWBG	0.23	0.42	-0.73
MCLA+	0.26	0.43	-0.46
LWMC	0.25	0.37	-0.47
DREC*	0.3	0.32	-0.51
LPMMC	0.31	0.41	-0.62
DISC_A*	0.16	0.31	-0.54
DISC_M*	0.2	0.33	-0.28
DISC_R*	0.18	0.32	-0.51
DICL_A	0.36	0.41	-0.53
DICL_M	0.18	0.37	-0.83
DICL_AG	0.23	0.39	-0.46
DIMC_A	0.25	0.36	-0.61
DIMC_M	0.18	0.39	-0.65
MSIM_C	0.22	0.35	-0.26
CVIC	**0.38**	**0.45**	-0.49
CNDC_C1	0.35	0.37	-0.46
CNDC_C2	**0.38**	0.42	-0.48
CNDC_I	0.34	0.37	-0.44
SFI*	0.24	0.38	-0.29
SPSL*	0.21	0.25	-0.44

CG*	0.32	0.39	-0.49
ResK*	0.25	0.4	-0.28
IPMMC	0.23	0.42	-0.51
NF-CCE*	0.23	0.26	-0.097
ETL-MSC*	0.3	0.35	-0.4
k-Means*	0.18	0.17	**0.97**
Ward Agglomerative*	0.18	0.17	**0.97**
mod-kNN	0.12	0.22	0.05
OPTICS	0.01	0.14	0.177
Spectral with Gaussian Transformation*	0.01	0.04	-0.39

Table 5.3: Clustering goodness scores for different clustering methodologies on malware samples coming from three different threat actors. The top 3 performing technique for a goodness score are in green and the best performing technique is bolded. The partitions from the top are late integration, hybrid integration, intermediate integration and the bottom are uni-modal clustering techniques, performed on early integration of the modal features. Techniques denoted with a * require the number of clusters as input.

The clustering goodness test presented several interesting results. First a hybrid integration method, CVIC, performs the best in terms of finding clusters that agree with the malware family labels. Its performance is closely followed by other hybrid integration techniques like the CNDC family of techniques. Each of these are intermediate integration, multi-view clustering techniques. It is interesting to note that all of these techniques rely on a diffusion process for combining the graphs into the view clusterings. Additionally, CVIC along with the CNDC family of techniques do not rely on knowledge of the number of families present in the malware. Thus, CVIC is able to group the malware samples along threat actor lines heterogeneous types of malware, without knowledge of how many families may be present in all of the malware samples.

Second, in every case, choosing to represent and cluster the malware samples with intermediate or late integration approaches as opposed to early integration gives better clustering results. This is a surprising result as there is nothing *a priori* that would suggest treating malware in a multi-view fashion would provide for better malware threat group identification. Furthermore, Silhouette does not give a good indication of a clustering method's performance across the multi-view techniques. When compared to early integration, uni-modal silhouette scores, the clusters found by the multi-view techniques have almost universally worse Silhouette scores indicating that they should have universally worse cluster assignments. However, the tests clearly show intermediate and late integration, multi-view clustering algorithms as being better able to find threat actor groups in the samples. Furthermore, there is also no trend in the Silhouette scores within the intermediate and late integration, multi-view techniques; more positive Silhouette scores do not indicate better clustering results for these techniques. So, the cluster goodness testing indicates that multi-view clustering performs the best at identifying meaningful threat actor labels from malware samples and that intermediate and late integration, multi-view techniques always outperform early integration with uni-modal techniques in this clustering task.

5.3.4 Analysis of Discovered Communities

Having demonstrated the usefulness of CVIC in finding threat actor groups from samples of malware data, I then analyzed the communities found by the CVIC method. As mentioned in the method section, after preforming network influence iterations, I clustered the final output as a bipartite graph with a method that does not require the specification of the number of clusters (Bi-Louvain [155]). As such, the method will obtain clusters of both the malware samples and the original clusters found from each view of the malware data. The method may also have a different number of clusters than number of threat actor groups. The following table, Table 5.4 displays the clusters found by the CVIC method with their membership of malware samples and view clusterings.

CVIC Cluster	Number of Malware Samples	Number of view clusterings
1	674	32
2	617	29
3	365	17
4	242	16
5	173	6
6	114	4

Table 5.4: Numbers of malware samples and clusters from each view of the malware samples in each of the final, CVIC-derived clusters.

The CVIC with a Bi-Louvain clustering produces roughly double the number of clusters as malware family labels. The found clusters also display heterogeneity in terms of size of the clusters with the largest cluster having six times as many malware samples and view clusterings as the smallest cluster. The differences in cluster sizes are also the same for each of the cluster constituents; the largest cluster in terms of malware samples is also the largest cluster in terms of view clusterings, and so on.

Since there is distinct heterogeneity in the CVIC-found clusters, I then analyzed the relationship between the found clusters and threat actor groups. The following figure, Figure 5.3, displays the relative number of each threat actor found in each found cluster

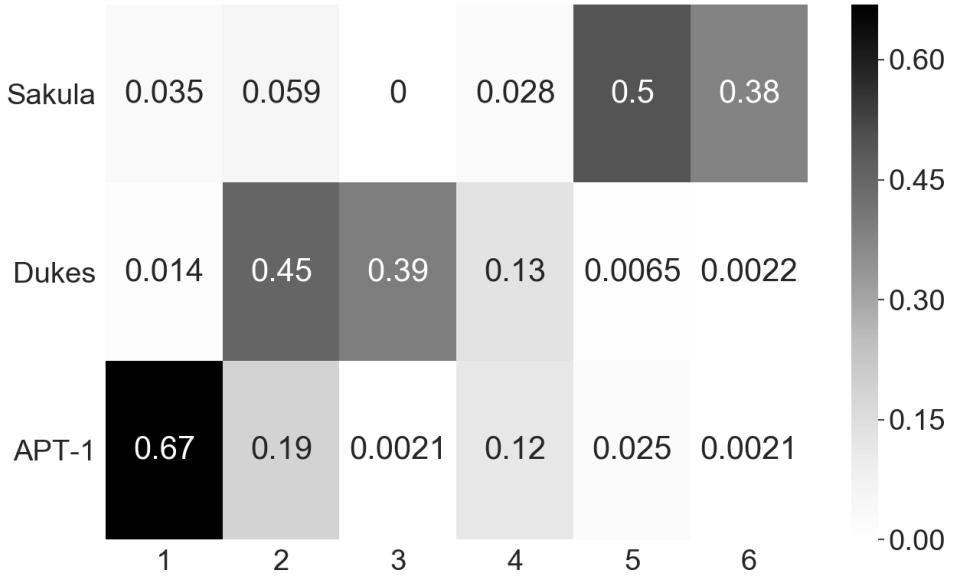


Figure 5.3: Fraction of each malware threat actor found in those clusters obtained by CVIC method.

The majority of APT-1 malware falls into the first, and largest cluster. Since APT-1 makes up most of the samples across all of the threat actors, it would stand to reason that it would dominant the larger clusters. More interestingly, the majority of samples between the Dukes and Deep Panda threat actors break into two clusters. In the case of Deep Panda, previous research on characterizing communities from malware samples found a similar result of their being actually two main communities of malware present in the Deep Panda threat group [36]. Thus, it may be possible that the Dukes and Deep Panda threat actors are more heterogeneous than the APT-1 in terms of their malware usage. This is an unusual result, since APT-1 and Dukes contain heterogeneous types of malware in its samples, while the Deep Panda samples are all from the Sakula family. This result may suggest that even though threat actors may use different tools, they may also share a lot of code between those tools, resulting in greater or lesser homogeneity in their malware. Also, this result may suggest that the Dukes and Deep Panda threat actors could actually be more than one threat actor. Finally, there is an outlier cluster in cluster number four. This cluster does not contain a predominance of any of the threat actors and may represent unusual malware samples or a different malware actor entirely.

Since the samples from the Dukes threat actor contains some malware family labels as identified in [45], I can analyze how the Dukes malware families distribute across the CVIC-discovered clusters. The following figure, Figure 5.4, displays the relative number of each the Dukes malware families found in each found cluster.

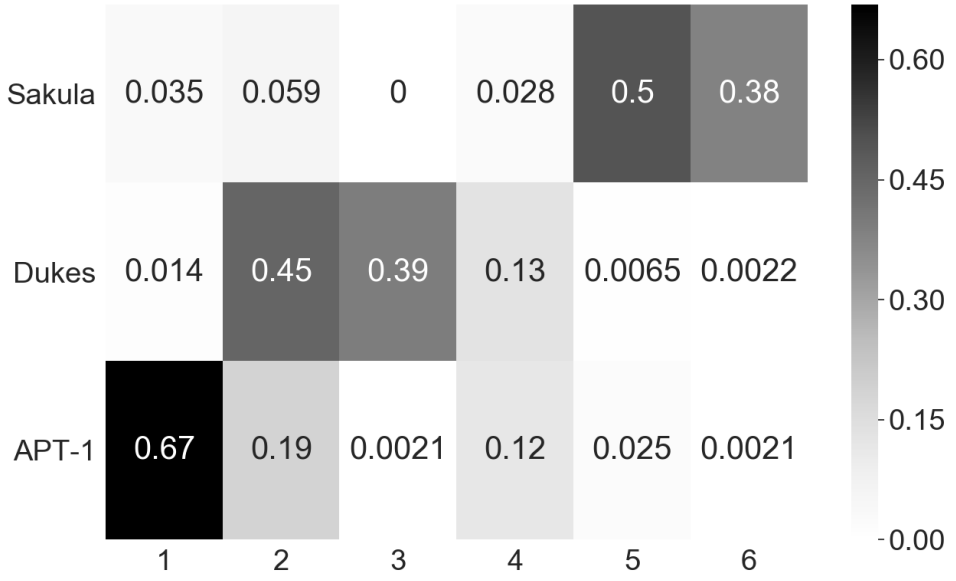


Figure 5.4: Fraction of Dukes malware families found in those clusters obtained by CVIC method.

Not unsurprisingly, most of the dukes samples fall within clusters 2 and 3 which contain most of the dukes samples. However, their distribution is not equal between clusters 2 and 3, with cluster 2 having most of the family-labeled Dukes malware samples. While its not clear why this division exists it may be related to when the samples were produced, with those in subgroup 3 coming from a later time period than those in subgroups 2, since the Dukes family labels come from a 2015 report on the Dukes threat actor [45] Furthermore, the one sample that belongs to the MiniDuke family fell into the outlier cluster (cluster 4) along with most of the PinchDuke samples. PinchDuke was an earlier family of malware used by Dukes which briefly had some code overlap with MiniDuke and was replaced by a near complete re-engineering of their malware tools [45]. So, this would suggest that subgroup 4 does indeed contain more of the anomalous samples across the different threat actors. Overall, it would seem given some of the family labels from the Dukes threat actor that the relationship between threat actors and the malware families that they employ can be a complex one.

Since the CVIC method also obtains clusters of the original view clusterings, I analyzed the relationship between the clusters of the different modes and the found clusters. The following figure, Figure 5.5, displays the fraction of each mode's clusters that are present in each of the found clusters.

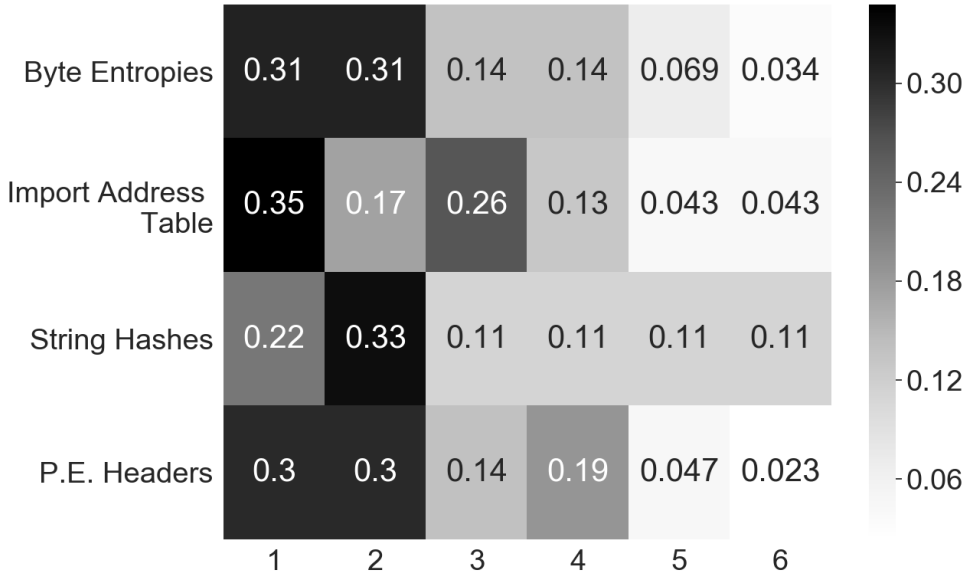


Figure 5.5: Fraction of each of the modes' clusters found in those clusters obtained by CVIC method.

Generally speaking, each mode's clusters are not equally distributed across all of the found clusters. The first two found clusters contain most of the clusters from all of the modes. This is not a surprising result, as the first two found clusters contain most of the malware samples across all of the modes. However, despite the preponderance of the first two found clusters, the distribution of view clusterings differs across the found clusters. For instance, both the clusters found from the import address table features and the clusters found from the string hashes have significantly different distributions of clusters across the found clusters than do those found using byte entropy and P.E. Header features. Given that found clusters also have different distributions of malware families, this result implies that there are differences in the modes of the features between the malware families. For example, cluster one contains mostly samples from the the APT-1 malware family whereas cluster two contains mostly samples from the Dukes malware family. These two found clusters also contain a lot of byte entropy and P.E. header clusters but differ significantly on the string hash clusters and import address table clusters. This may indicate that even though both of these malware threat actors use many different types of malware that could be similar in some respects, that their malware can still be distinct in other areas like their use of strings or import address table entries.

Overall, while the found clusters do generally follow the the threat actor groups there are some subtle differences between the two labelings as well. For two of the three threat actors, their found clusters actually tend to break into two main clusters which could indicate some distinct heterogeneity in those threat actors. Also, not all of the clusters found in each of the modes distribute the same way across all of the found clusters. This difference indicates that there are differences between the threat actor groups within the modes as well as across the modes.

5.3.5 Analysis of Multi-view Nature of Malware

One of the main findings in the previous section was that clustering the malware samples using intermediate and late integration, multi-view approaches always produced better clusters relative to the malware threat actor labels than representing the malware as uni-modal through an early integration approach. To investigate why this might be, I first investigated how well each mode's clusters relate to the threat actors. The following table, Table 5.5 presents the clustering goodness scores for each of the view clusterings, as found in the previous analysis by the mod-kNN procedure, relative to the threat actors.

view	ARI	AMI	Silhouette
Import Address Table	0.19	0.30	-0.85
Byte Entropies	0.17	0.26	-0.57
String Hashes	0.16	0.21	-0.57
P.E. Headers	0.13	0.22	0.19

Table 5.5: Cluster goodness results for each of the Berlin and Saxe feature sets, or modes, of the malware data.

For each view of the malware data, no particular view is especially good at clustering relative to the threat actors, however, the Import Address Table features do perform as good as the early integration clustering results. Thus, the Import Address Table features seem to have more relation to the threat actors than do any of the other sets of features. Furthermore, by concatenating all of the modes together in an early integration approach, the particular patterns of the Import Address Table become muddled with all of the other modes' patterns. In order to investigate this idea, I then analyzed how similar the clusters from each view are to every other mode. The following set of figures, Figure 5.6, display the ARI and AMI values for each set of modes.

(a) ARI

(b) AMI

Figure 5.6: Comparison of the clusters found within the different modes of the features.

From the ARI and AMI values between the view clusterings, it is clear there is a fair amount of overlap between the cluster assignments of each of the modes. Most of the modes have ARI values greater than 0.4 and AMI values greater than 0.5 with every other mode, with the Byte Entropy features being the only exception. These high values indicate that the clusters for the

malware samples between each of the modes are reasonably similar. However, the cluster assignments are not identical. This would indicate that using intermediate or late integration approaches for malware allows for exploitation of the differences in cluster assignments between the modes in order to produce better, overall cluster assignments. So, while not all modes of the Berlin and Saxe malware features are as useful as the others in identifying the possible threat actors from their malware usage, analyzing the data with intermediate and late integration approaches allows for better threat actor characterization because these methods can better exploit the differences in the clusters between the modes.

5.3.6 Analysis of User-Set Parameters of CVIC

I now turn to analyzing how robust the CVIC method is to changes in the user set parameters of the number of iterations and network influence strength, α. Beginning with the number of iterations to run the CVIC algorithm, I held the network influence strength value constant at $\alpha = 0.9$ and ran the algorithm for a number of iterations and recorded the AMI and ARI values at that iteration. The following plot, Figure 5.7 displays the ARI and AMI values as a function of the number of iterations of the algorithm.

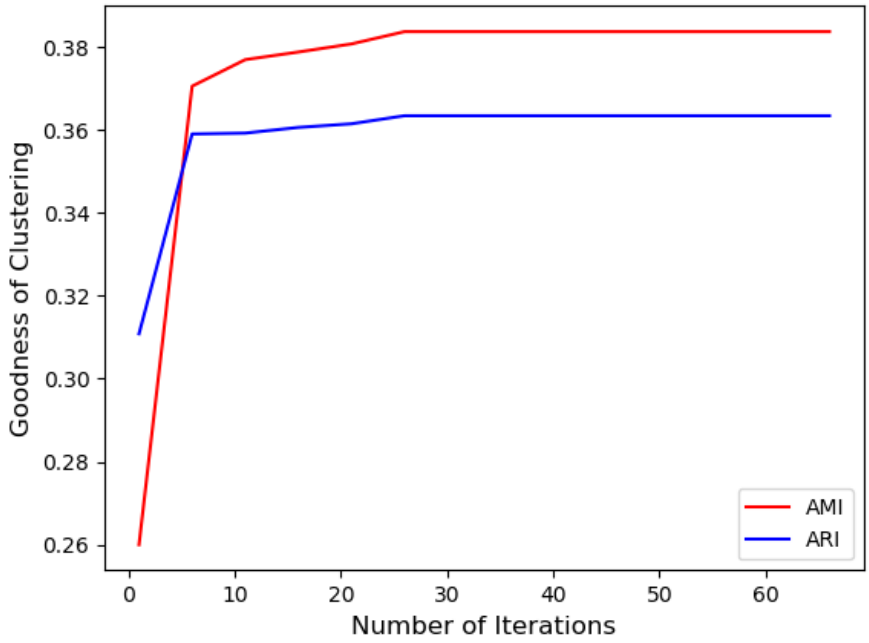


Figure 5.7: Iterations of the CVIC method versus the clustering goodness relative to the malware family labels

As can be seen from the plot, the algorithm gets within good values of AMI and ARI in only 10 iterations. Furthermore, the algorithm stabilizes to its best AMI and ARI values by around 30 iterations. Thus, the algorithm not only converges to reasonable cluster labels quickly, but is also stable with the number of iterations.

Turning to the network influence parameter, α, I ran the CVIC algorithm for various values of α and fixed the number of iterations at 30. Most diffusion based processes, to include the social influence one I use in this work, typically have higher values of network influence ($\alpha >= 0.7$) [146], [136], [51]. So, I varied α from 0.5 to 0.99. The following plot, Figure 5.8, displays the goodness of clustering results for the various α values.

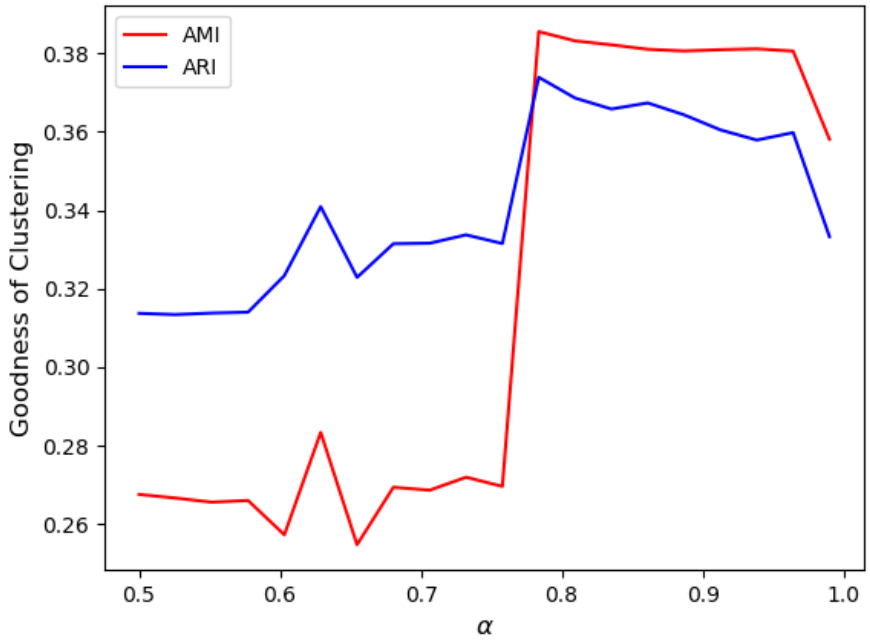


Figure 5.8: α values of the CVIC method versus the clustering goodness relative to the malware family labels

From the figure, when α is within normal ranges of 0.75 to 0.95, the resulting AMI and ARI values remain stable with less than a 0.02 variation in their values. Thus, the CVIC algorithm is also robust to the selection of the network influence strength parameter, α, as long as the value is within a reasonable range of 0.75 to 0.95. These results show that while a user must supply some parameters for CVIC, the algorithm is robust to the user input and can converge to reasonable cluster results quickly.

5.4 Discussion

This chapter presents the discovery and characterization of heterogeneous malware samples into threat actor groups from a multi-view, unsupervised learning context. In this chapter I used the static malware features originally proposed by Berlin and Saxe in [111] to identify families of malware from three different families: APT-1, Deep Panda, and Dukes. The results of the empirical tests first demonstrated that finding communities in malware samples using intermediate and late integration, multi-view approaches, instead of early integration approaches, always resulted in better threat group characterization. I believe this finding is a result of malware development

more generally. Since many makers of malware share aspects of code between their different malware tools or only make small tweaks to a particular module of the malware to evade anti-malware products, it should then follow that having features and algorithms that are sensitive to the changes in only certain aspects of a malware sample should produce better results. So, a key finding of of this chapter is that the characterization of malware samples into threat groups should be done from a late or intermediate integration, multi-view approach.

Secondly, the hybrid integration algorithm, CVIC, outperforms all the other algorithms in placing malware samples into their correct threat actor groups with a set of malware samples from three different threat actors. Hybrid techniques like CVIC bridge the gap between the two lines of methodological research in multi-view clustering by using cluster labels specific to each view, but also uses diffusion across graphs constructed from each view. Thus, by using advances from both lines of research I was able to design an algorithm that produces superior results in finding clusters of malware from multi-view features.

The results of this case study chapter present several interesting avenues for future research. First, In this chapter I used a select set of static features derived from malware samples. It is well known that static features for malware identification can suffer from various forms of obfuscation applied to the malware [62]. It would be interesting to see if more and wider varieties of features taken from the malware samples, to include features from dynamic analyses, may be even better at identifying malware threat groups from a multi-view data perspective. Second, it would also be interesting to investigate a wider variety of malware threat actor groups of different types of samples of malware (i.e. Trojans versus viruses, etc.) to see if the the multi-view clustering approach still recovers useful groups of threat actors. In fact, while the experiments in this chapter did produce reasonable threat actor groups in an unsupervised way, no particular algorithm performed particularly well at finding the threat actor labels from the malware samples. It would seem that characterizing threat actors, and not just malware families, from malware samples remains a challenging problem. It would be interesting to test whether incorporating more social-based data like observed collaboration networks might help with overcoming the limitations of just using the malware samples. Finally, I also intend to investigate variations on the CVIC algorithm to include network influence thresholds and the use of weighted averaging during the mode-combining step.

Chapter 6

Concluding Remarks

Social interactions among humans are a complex affair. They are blend of many types of content and connections between people that occur in myriad, heterogeneous, and dynamic ways. This complexity can give rise to many important social phenomena and also be a source of great difficulty for understanding those social interactions. This complexity has only increased in recent years. With the advent of online social media, there is now an unprecedented level of detail that can be used to describe all of the many social interactions that shape our world. This level of detail and volume of available data can lead to greater insights into understanding social interactions, but also requires more computational tools to handle the volume and complexity of the data. In this thesis, I have sought to develop techniques and methods to address a particular aspect of understanding complex, social-based data, that of clustering multi-view, social-based data.

Many social interactions naturally give rise to multiple modalities of data. Publishing papers gives rise to the actual text of the papers as well as citation or co-authorship networks. Organizations give rise to multiple types of in person and electronic interactions between members of the organization. Online social media sites give rise to user-to-user interactions as well as users' content. All of these different modalities offer views — often, incomplete views — into the fundamental relationships between the actors engaging in the social activities. With this insight, I have sought to take advances in the field of multi-view, or multi-modal, unsupervised machine learning and apply those advances to better understand multi-view, social-based data. In the process of investigating the various techniques from the field, I have proposed a new hybrid paradigm of techniques, both proposed and empirically demonstrated the effectiveness of some multi-view clustering techniques on benchmark, social-based data, and demonstrated these techniques on real-world, social-based clustering scenarios. Finally, using the findings and techniques developed in the early chapters of this work, I apply multi-view clustering to two different clustering investigations of real-world data, both of which show interesting results with respect to exploring the data. So, this work builds upon two important lines of research. First, it is the first work to use multi-view clustering to understand social-based data. Second, it is the first work to empirically assess techniques from multi-view clustering on social-based data. In so doing, this work both advances a better means of clustering social-based data as well as highlighting important assumptions and characteristics that have been embedded into much research on clustering multi-view data. It is hoped this research can aid practitioners analyzing social-based data

to better analyze their data as well as those who develop new techniques to consider additional aspects of real-world data which occur with social-based data scenarios when developing their techniques.

6.1 Contributions

The overall aim of this thesis was to investigate, develop, and employ multi-view clustering for social-based data. As such, this is the first work to investigate multi-view clustering on generic, social-based data. There has been a wealth of research on multi-view clustering for data such as image data or genetic data, as well as a wealth of research on clustering complex networks like multi-layer or multiplex networks. However, there has been relatively little research in the way of clustering social-based data beyond attributed graphs — which only consider one network mode with one non-network mode — which could could consist of any number of network and non-network views of the data. In this work, I have shown techniques for this more general form of multi-view clustering and demonstrated its effectiveness using both various types of multi-view, social-based benchmark data sets and real-world case studies.

In the second chapter, I performed the first comprehensive, empirical investigation of several techniques from the different paradigms of multi-view clustering as well as proposed a hybrid paradigm for multi-view clustering. Many previous works on multi-view clustering typically tested their methods only one type of social-based data, like only publication networks or only multiplex networks, if they tested on social-based data at all. In this chapter, I used benchmark data sets from multiple scenarios of social-based data that give rise to multi-view data. The results of this comprehensive testing demonstrated that while some techniques can work quite well for some social-based data scenarios, they often work very poorly for other scenarios. In other words, there is no one technique that is always good for every social-data based scenario that gives rise to multi-view data. Furthermore, there are also some key differences between image or genetic data and data which arises from human, social interactions. Some of these differences are described below:

- **Network and Non-network data**: Most non-social-based data scenarios give rise to non-network data. And, those methods that have been designed for social-based data largely only consider the scenarios that give rise to network views of data. However, for social-based data, there are often a combination of non-network and network views which poses distinct challenges for multi-view clustering. For example, the network views will often have topologically dissimilar graphs to the graphs learned from the non-network views of the data.
- **Topologically difficult view graphs**: When working with non-social based, multi-view data (and, even some social-based data scenarios) one often gets to pick the graph representations of the different views of the data. For example, many image and genetic multi-view clustering techniques use k-Nearest Neighbor graphs for each view of the data. This allows for the graphs to have convenient properties in terms of connectedness and having similar local structures across the entire graph. Many naturally arising networks, however, do not have these computationally-convenient graph properties and, furthermore, may not display them even if a similarity graph is learned from the networks. For example,

it is not uncommon to have some actors in a social-based scenario that do not participate in a certain view, and so that view will always have isolates in its view graphs. So, it would seem social-based data naturally gives rise to difficult topologies of graphs which other types of data scenarios do not.

- **Selection of views for multi-view clustering**: In some data scenarios, it is clear what views should be used in multi-view clustering and what the cluster labels should be. For example, in image clustering, it is generally clear that clusters of images should be all images of same thing and that different ways of comparing the images (i.e. filters or similarity measures) all contribute to this cluster structure. The selection of appropriate views that could contribute to a desired cluster structure is much less clear for social-based data. It is generally well known in social science that individuals often belong to many different communities of different scales. So, all of the observable interactions between people may not arise from the same latent cluster structure, and its not always clear which interactions should be incorporated into a multi-view clustering and which should not be.

Additionally, I also proposed a new paradigm for clustering multi-view, social-based data. Existing techniques, with very few exceptions, break into three paradigms for clustering multi-view data: early, intermediate, and late integration techniques. In my work, I proposed a suite of new techniques that bridged the intermediate and late integration paradigms. These techniques use both the clusters from each of the views, as well as intermediate paradigm quality functions and data representations to create new hybrid paradigm techniques. Empirical testing of these techniques revealed that a hybrid paradigm of clustering multi-view data shows promise for developing superior to techniques for clustering multi-view, social-based data.

In the third chapter, I used findings from the empirical investigations of the second chapter to develop a new technique for multi-view clustering of social-based data called Multi-view Modularity Clustering (MVMC). This technique uses recent advances in modularity-based clustering to iteratively determine the best clusters for the multi-view data from an intermediate integration paradigm and set optimal parameters for the method. The MVMC method not only shows strong performance across a range of multi-view social-based data scenarios — occasionally achieving the best performance of any method considered in the investigation — but also provides additional insight into the data through its optimally-determined view weights and the number of iterations the method takes to reach convergence. Furthermore, empirical testing of the various modularity functions used in the chapter demonstrated that modularity can be a good evidence-based measure to judge the cluster structure in a multi-view, clustering scenario. So, the MVMC method provides a novel and consistently good means of clustering multi-view, social-based data that can handle partially incomplete views and large multi-view data sets.

Given the findings of the second and third chapter, there are some evidence-based recommendations for assessing the appropriateness of the different classes of techniques investigated in this work. Since different techniques by their design emphasize different aspects of what it means to have a cluster in the data, they also have different data scenarios where they are more or less appropriate. In particular, in this work, there were five main aspects of the data that were found to distinguish between the techniques in terms of their performance. The following figure, Figure 6.1, displays a qualitative summary of the different types of techniques and the particular aspects of the data for when they perform well or poorly.

Class of Techniques	Partially complete views or views wherein not all actors interact (view graph with disconnects; Typically more than 30% of objects being isolates per view or having objects be an isolate in more than one view are issues.)	Differences between data topologies of views (view graphs with widely varying densities on the order of 100 times the difference, different components, assortiavities, etc.)	Differences between cluster structures of views (i.e. having ARI <0.25 between the view clusterings)	Large number of entities that need to be clustered (i.e. >10,000 objects)	High cluster to number of entities ratio (i.e. ≈ 0.08 or greater. Or more than 8 clusters per 100 objects)
Late integration techniques (i.e. CSPA, BGPA)			−	+	
Hybrid Integration, direct integration (i.e. DISC, DICL)					
Hybrid Integration, diffusion integration (i.e. CVIC)		+		−	
Intermediate Integration, matrix or tensor factorization (i.e. ETL-MSC, NF-CCE)			+	−	+
Intermediate integration, spectral (i.e. SPSL)	−			−	
Intermediate integration, modularity (i.e. MVMC)	+	+	+	+	

Figure 6.1: The following figure qualitatively depicts different aspects of the data and how different technique are able to handle those different aspects of the data, as determined by the empirical testing done in this work. Those marked with a red minus sign are when that particular class of techniques is especially poor for when the data has a particular aspect and vice-versa for the green plus sign. If a class of techniques was not found to be particularly good or poor for a given aspect of the data, then no symbol is given. The modularity-based, intermediate integration techniques tend to have superior performance across a wide range of data characteristics.

Generally, the modularity-based, intermediate integration techniques like MVMC tend to have superior performance across a lot the aspects of data that arise from social-based scenarios. In particular, social-based features like having actors which do not interact within certain views (e.g. users which never retweet or use hashtags) of the data is only really overcome through optimizing the clustering procedure on a network modularity-based function. Other functions, like spectrally-based functions, can suffer in these types of data scenarios as they rely on the spectral properties of the view graphs' Laplacians, which will necessarily be effected by any disconnects in that graph. So, when real-world, social-based scenarios give rise to partially complete views — as they often do — using a graph Laplacian-based model can have issues with recovering the clusters. It should also be noted that many techniques are also not scalable to the size of the data that occurs in social-based data scenarios. Many social-based data scenarios, especially those occurring on social media, will necessarily have a need for clustering thousands to hundreds of thousands of entities. However, many techniques, especially in the hybrid and

intermediate integration paradigms, rely on dense matrix-matrix multiplications as well as matrix factorizations. Both of these operations can be computationally expensive for large data. Overall, the evidence-based findings from this work indicate that not all techniques are created equal when it comes to clustering social-based data; social-based data can have many different aspects and any given clustering technique naturally encodes bias about the nature of good clusters, which may not be amenable to certain real-world aspects of the data.

In the fourth chapter, I provide the first exploratory clustering analysis of hashtag usage on Twitter during the COVID-19 pandemic. Hashtags are a social media innovation that allows users to find and participate in discussions of interest. So, clusters of hashtags can often give unique insight into topical focal points for social media users. In that chapter, I performed the first ever multi-view clustering of hashtags from social media data. The use of the MVMC technique not only demonstrated that multi-view clustering can produce meaningful topical clusters of hashtags, but can do so with partially complete views, noisy data, and large data sets (on the order hundreds of thousands of hashtags). Additionally, analysis of the hashtags clusters over the course of the pandemic showed not only temporal patterns in hashtags usage, but also some distinct insights into how different users or groups use certain topics of hashtags for different purposes. For example some topical areas of hashtags are used to draw users participating in the online discussions on the COVID-19 pandemics to other calamities, like the Syrian Civil War. So, techniques from this thesis, like MVMC, can be used on large, noisy, real-world multi-view social-based data scenarios to produce meaningful clusters.

Finally, in the fifth chapter, I used multi-view clustering to perform a novel community analysis of malware samples. In this chapter, I demonstrated that using a multi-view clustering approach on features derived from static analyses of malware samples could provide meaningful clusters, especially with regards to the providence of the malware samples. My research also demonstrated that taking a multi-view clustering approach, regardless of the technique, provided better clustering results relative to the threat actors that produced the malware samples, than did any early integration or traditional approach to clustering malware samples. So, the techniques developed in this thesis can even be applied to non-traditional sources of social-based data, like malware samples.

6.2 Guidance for Practical Usage of Multi-view clustering on Social-Based Data

Over the course of the research for this thesis, some consistent practices have emerged when using multi-view clustering on social-based data. These practices are summarized here for practitioners to keep in mind when using multi-view clustering on social-based data.

- **Views of the data should all relate to the same cluster structure** The selection of views used in clustering of the data should all relate to the same cluster structure of the data. Not all views of the same underlying objects will give rise to the same cluster structure. For example, people's online communities may not be the same as their in person communities, and so trying to recover one community structure from both online and in-person data may result in a poor community structure. So, for each view of the data, it is important to either

know or have a reasonable mechanism by which that view relates to an underlying cluster structure of the data, and that each view relates to the same cluster structure. Put another way, the cluster structure present in each view of the data should all be, in some way, a result of the same latent cluster structure. Thus, when using multi-view data, one should ask themselves whether each view of the data relates to the same cluster structure and posit how each view of the cluster structure relates to the same, latent cluster structure

- **Whether to use the raw networks or similarity graphs for network views of the data** As has been mentioned throughout the thesis, multi-view social-based data can often be in both network and non-network forms. Since many intermediate integration techniques (including the ones developed in this thesis) require all views of the data to be modeled as a graph, a natural question arises: does one convert the network views of the data not a similarity graph, as one would do with the non-network modes, or not? In this thesis I have used both approaches of converting the networks into similarity graphs and not converting them to similarity graphs. Generally, I have found that it is preferable to leave the networks in their raw format if a link in a network represents a significant, positive social interaction, and otherwise convert it into a similarity graph. For example, on Twitter, retweeting another user's tweet does not typically represent a significant social interaction between the user which posted the original tweet and the user which retweeted that tweet. So, in this case, a more useful edge for clustering purposes would be one which models how often two users retweet the same other users, rather than the users they are retweeting themselves. So, the research conducted in this thesis indicates that one should convert network views of the data to similarity graphs when links in the network do not represent significant social interactions between the nodes.
- **Always check graph statistics of the view graphs** When performing multi-view clustering, especially with intermediate integration techniques that rely on graph representations, one should always check the graph statistics of the view graphs. In general, from Chapter Two and chapter Four, it was empirically observed that graphs which presented a less defined cluster structure than the other view graphs were also less useful for finding clusters in the multi-view data. So, if one or more views differ substantially in the number of connected components with the other views, that view, in its current graph form, will likely not contribute much to the multi-view clustering. Additionally, if it is feasible to do so, calculating the average clustering coefficient of the view graphs can also give an indication of how well the graphs are modeling a cluster structure, with a higher coefficient correlating to a better cluster structure. If a particular view graph presents significantly more components than any of the other views, it may be worth excluding this view from the multi-view clustering or otherwise transforming the graph.
- **Selection of the technique to do multi-view clustering** For some social-based data scenarios, certain techniques do tend to work well for clustering the multi-view data. Table 2.19 contains a list of these techniques and what social-based data scenarios they perform well on. However, as has been shown in this thesis, no technique is always the best across all of the social-based, multi-view data scenarios. So, when in doubt of the technique to use for a particular social-based data scenario, it is best to use techniques which are broadly good, if not always the best, like MVMC or CVIC.

- **Evaluation of the found cluster structure** After performing the multi-view clustering — if the technique used allows for it — evaluate the strength of the cluster structure and importance of the different views in producing the multi-view cluster structure. For example, network modularity-based methods can use the value of the modularity of the final multi-view clustering to evaluate how strong the multi-view cluster structure is (i.e. having a larger network modularity value, weighted or otherwise, is positively correlated with their being a stronger cluster structure present within the data). Also, some methods like MVMC provide view weights which can be used to assess the relative importance of the different views in determining the multi-view cluster structure. Overall, these evidence-based measures can be used to determine if the cluster structure is strongly present in the data and thus should be trusted for analysis, and what views mostly contribute to the cluster structure, which is also important for analysis of the data.

6.3 Limitations and Future Work

There are some salient limitations for this research and several avenues for future research. First, the clustering in this work has focused on a scenario where all of the views of the data relate to the same, single latent cluster structure for the data. This is, in some ways, a simplification of the clusters that can arise from social-based data; social-based data scenarios, and the actors which give rise on them can have different cluster structure for different views of the data. This is a natural extension of the social science observation that we, as social actors, can often belong to many different communities depending on the social context [94], [137]. Additionally, the same set of data could contain more than one valid clustering. Some recent research into multi-resolution network clustering has highlighted that clustering structure can exist at multiple resolutions and that some clustering structures can be stable across a range of resolutions values [138], [74], [24]. This is all to say that a given set of data may have more than one valid clustering structure and that the different clustering structures can represent the presence of different scales in the data (i.e. clustering on geographic data could yield multiple multiple valid clusterings like local county, state, or country, each of which differ in scale). So, an avenue of feature research would be to expand the method to ascertain if there is more than one valid clustering structure and recover those clustering structures.

Also, operating in a multi-view clustering paradigm means that every agent is at least capable of interacting in every view. For some social-based scenarios, this may not be a reasonable assumption (i.e. some individuals may not have access to certain social media sites and so cannot interact within them and the views they generate, when combined with real-world views of the actors). So, an interesting avenue for future research would be to develop techniques that can address the multi-view clustering paradigm limitations. For example, within the world of complex network clustering, there are techniques which can operate on multi-layer networks, where each layer does not need to have the same clusters (i.e. [93], or [101]). It would be beneficial to have similar techniques that can work on network and non-network views of the data to produce more than one latent clustering across multiple views of the data, and could even determine which views of the data are responsible for which latent clusterings.

Second, many of the conclusions and results from this research are based on empirical find-

ings on only social-based data. I have endeavored in this research to compare techniques on a broad array social-based data scenarios, but they are certainly not all of the possible scenarios that exist. Furthermore, no testing on non-social based data was done in this research. So, while I used techniques that were not explicitly designed for social-based data in this thesis it is not clear if the reverse can be done, and those techniques designed for social-based data be used on non-social-based data like images or genetic data. So, another promising avenue of future research is to see if the techniques presented in this thesis can work in more data scenarios in both social-based and other types of data scenarios.

Third, there are some limitations with the two main methods presented in this work, CVIC and MVMC, which also apply more broadly to intermediate integration techniques. The main limitation is how the graph for each of the views are constructed. In this work, I have used the modularity kNN procedure [110], [25], [26], and simple symmetric graphs as the means of learning graphs for each of the views of a data set. While these did work for this research, they are not guaranteed to work generally or even find the best fit graph for a view of the data. So, a promising avenue of future research would be to find a scalable means of learning a best graph of the data such that the graph optimally draws out the cluster structure present in the data. Furthermore, it would also be interesting to see if the graph earning step could be included in the clustering iterations of techniques like CVIC and CMIC, such that one could both find optimal clusters and optimal graph representations of the data in tandem, as has been done with in modularity kNN procedure and spectral clustering or non-negative matrix factorization procedures [125], [150], [84]. So, finding a better means of learning graphs for the different views of any given data set is promising avenue of future research for multi-view clustering of social-based data.

Finally, while showing some promise in clustering multi-view, social-based data, using information from across the different multi-view clustering paradigms has not been fully investigated. In this work I presented some techniques which use both intermediate and late paradigm information in their clustering. While some techniques did preform well, many also did not. So, while these techniques have access to more information, they do not always produce better results. So, while there is some promise in the idea of using hybrid paradigm techniques, how these techniques should be designed and what aspects of the different paradigms should be incorporated is still very much a work in progress. Thus, an avenue of future research is to investigate what information from the various stages and paradigms of multi-view clustering is actually useful to the overarching goal of finding the latent clusters in multi-view data.

In many respects, unsupervised learning on multi-view, social-based data is still in its infancy. Even the more robustly researched areas of multi-layer and multiplex network clustering are still active areas of research that are producing new techniques and expanding their scope of applications to practical problems. There remain many interesting problem domains for the field including how to handle various data types, how to combine information for those various data types, and how to appropriately and ethically select different views of data, among many others. So, I sincerely hope the research done in this work provides a meaningful advancement for researchers to better understand social-based data. And, I very much look forward to seeing the field expand and produce meaningful insights on the nature of social interactions through their many different data views. In a world that is becoming increasingly digitized and with social media becoming a *sine qua non* of modern life, the need to understand complex social interactions will only increase.

Appendix A

Additional Modularity for Multi-view Clustering Results

This appendix contains the detailed performance results for all of the modularity maximization-based techniques proposed in Chapter 2 and tested in Chapter 3. These results include both late and intermediate paradigm multi-view clustering techniques. For each method, the performance results are detialed for every benchmark data set and all of the views of the data sets.

Data set	View	Modularity of clustering for each view	Modularity of ground-truth clustering for each view	Difference between Modularities	Correlation between ground-truth ARI and clustering Modularity over optimization steps
Cora	network	0.507	0.643	-0.136	0.983
	text	0.502	0.343	0.159	
CiteSeer	network	0.579	0.551	0.028	0.98
	text	0.489	0.324	0.165	
Flickr	network	0.155	0.133	0.022	0.949
	attributes	0.444	0.42	0.024	
BlogCatalog	network	0.261	0.233	0.028	0.959
	attributes	0.499	0.5	-0.001	
Wiki	network	0.637	0.613	0.024	0.996
	text	0.779	0.587	0.192	
3Sources	BBC -text	0.534	0.492	0.042	0.983
	Reuters -text	0.532	0.504	0.028	
	Guardian -text	0.531	0.494	0.037	
AUCS	lunch	0.625	0.518	0.107	0.999
	facebook	0.284	0.209	0.075	
	coauthor	0.739	0.673	0.066	
	work	0.434	0.351	0.083	
	liesure	0.481	0.471	0.01	
Football	list	0.503	0.465	0.038	0.988
	text	0.132	0.119	0.013	
	follows	0.43	0.414	0.016	
	mentions	0.48	0.611	-0.131	
	retweets	0.555	0.629	-0.074	
Olympics	list	0.513	0.477	0.036	0.986
	text	0.321	0.279	0.042	

	follows	0.477	0.458	0.019	
	mentions	0.491	0.798	-0.307	
	retweets	0.639	0.842	-0.203	
Politics Ire	list	0.447	0.443	0.004	0.984
	text	0.232	0.219	0.013	
	follows	0.363	0.356	0.007	
	mentions	0.393	0.526	-0.133	
	retweets	0.538	0.66	-0.122	
Politics UK	list	0.576	0.573	0.003	0.999
	text	0.194	0.199	-0.005	
	follows	0.397	0.397	0	
	mentions	0.278	0.275	0.003	
	retweets	0.352	0.335	0.017	
Rugby	list	0.672	0.555	0.117	0.996
	text	0.303	0.264	0.039	
	follows	0.529	0.443	0.086	
	mentions	0.528	0.547	-0.019	
	retweets	0.584	0.586	-0.002	

Table A.1: Intermediate Paradigm Modularity Maximization Clustering (IPMMC) results for all of the multi-view, social-based, benchmark data sets. Performance results are detailed for each data set and each view of each data set.

Data set	Mode	Modularity of clustering for each view	Modularity of ground-truth clustering for each view	Difference between Modularities	Correlation between ground-truth ARI and clustering Modularity over optimization steps
Cora	network	0.111	0.508	-0.397	0.999
	text	0.83	0.116	0.714	
CiteSeer	network	0.189	0.402	-0.213	1
	text	0.814	0.098	0.716	
Flickr	network	0.659	0.048	0.611	0.999
	attributes	0.0964	0.4707	-0.3743	
BlogCatalog	network	0.036	0.0899	-0.0539	0.993
	attributes	0.698	0.2688	0.4292	
Wiki	network	0.654	0.279	0.375	0.971
	text	0.562	0.299	0.263	
3Sources	BBC -text	0.273	0.127	0.146	1
	Reuters -text	0.048	0.034	0.014	
	Guardian -text	0.119	0.045	0.074	
AUCS	lunch	0.667	0.418	0.249	1
	facebook	0.556	0.384	0.172	
	coauthor	0.75	0.672	0.078	
	work	0.455	0.3	0.155	
	liesure	0.423	0.391	0.032	
Football	list	0.335	0.344	-0.009	0.987
	text	0.017	0.0194	-0.0024	
	follows	0.822	0.368	0.454	
	mentions	0.797	0.686	0.111	
	retweets	0.59	0.516	0.074	
Olympics	list	0.352	0.358	-0.006	0.998
	text	0.078	0.0585	0.0195	
	follows	0.805	0.379	0.426	

	mentions	0.781	0.701	0.08	
	retweets	0.753	0.746	0.007	
Politics Ire	list	0.0635	0.059	0.0045	0.999
	text	0.161	0.154	0.007	
	follows	0.573	0.477	0.096	
	mentions	0.496	0.472	0.024	
	retweets	0.506	0.5	0.006	
Politics UK	list	0.503	0.5	0.003	0.999
	text	0.122	0.119	0.003	
	follows	0.538	0.5203	0.0177	
	mentions	0.492	0.485	0.007	
	retweets	0.517	0.499	0.018	
Rugby	list	0.589	0.45	0.139	0.999
	text	0.0202	0.0186	0.0016	
	follows	0.694	0.465	0.229	
	mentions	0.773	0.579	0.194	
	retweets	0.768	0.575	0.193	

Table A.2: Late Paradigm Modularity Maximization Clustering (LPMMC) results for all of the multi-view, social-based, benchmark data sets. Performance results are detailed for each data set and each view of each data set.

Data set	Mode	Modularity of clustering for each view	Modularity of ground-truth clustering for each view	Difference between Modularities	Correlation between ground-truth ARI and clustering Modularity over optimization steps
Cora	network	0.713	0.119	0.594	0.789
	text	0.252	0.483	-0.231	
CiteSeer	network	0.813	0.099	0.714	0.999
	text	0.194	0.387	-0.193	
Flickr	network	0.106	0.449	-0.343	0.999
	attributes	0.644	0.056	0.588	
BlogCatalog	network	0.074	0.268	-0.194	0.522
	attributes	0.653	0.101	0.552	
Wiki	network	0.356	0.312	0.044	0.997
	text	0.78	0.336	0.444	
3Sources	BBC -text	0.038	0.102	-0.064	0.706
	Reuters -text	0.101	0.147	-0.046	
	Guardian -text	0.12	0.135	-0.015	
AUCS	lunch	0.629	0.517	0.112	1
	facebook	0.498	0.412	0.086	
	coauthor	0.703	0.644	0.059	
	work	0.522	0.439	0.083	
	liesure	0.594	0.523	0.071	
Football	list	0.324	0.322	0.002	-0.356
	text	0.119	0.308	-0.189	
	follows	0.351	0.297	0.054	
	mentions	0.311	0.524	-0.213	
	retweets	0.328	0.366	-0.038	
Olympics	list	0.355	0.403	-0.048	0.638
	text	0.343	0.422	-0.079	
	follows	0.271	0.396	-0.125	
	mentions	0.317	0.717	-0.4	

	retweets	0.332	0.537	-0.205	
Politics Ire	list	0.367	0.434	-0.067	0.999
	text	0.283	0.356	-0.073	
	follows	0.176	0.331	-0.155	
	mentions	0.231	0.449	-0.218	
	retweets	0.236	0.43	-0.194	
Politics UK	list	0.321	0.449	-0.128	0.999
	text	0.4505	0.465	-0.0145	
	follows	0.298	0.418	-0.12	
	mentions	0.311	0.391	-0.08	
	retweets	0.319	0.441	-0.122	
Rugby	list	0.106	0.2801	-0.1741	0.979
	text	0.5003	0.46	0.0403	
	follows	0.0869	0.254	-0.1671	
	mentions	0.105	0.454	-0.349	
	retweets	0.107	0.289	-0.182	

Table A.3: Direct Integration Modularity Clustering (Additive) results for all of the multi-view, social-based, benchmark data sets. Performance results are detailed for each data set and each view of each data set.

Data set	Mode	Modularity of clustering for each view	Modularity of ground-truth clustering for each view	Difference between Modularities	Correlation between ground-truth ARI and clustering Modularity over optimization steps
Cora	network	0.871	0.676	0.195	0.999
	text	0.925	0.703	0.222	
CiteSeer	network	0.898	0.586	0.312	0.999
	text	0.934	0.581	0.353	
Flickr	network	0.838	0.662	0.176	0.996
	attributes	0.569	0.514	0.055	
BlogCatalog	network	0.763	0.597	0.166	0.999
	attributes	0.593	0.607	-0.014	
Wiki	network	0.892	0.768	0.124	0.999
	text	0.943	0.687	0.256	
3Sources	BBC -text	0.537	0.518	0.019	0.831
	Reuters -text	0.536	0.522	0.014	
	Guardian -text	0.585	0.514	0.071	
AUCS	lunch	0.685	0.571	0.114	1
	facebook	0.533	0.404	0.129	
	coauthor	0.732	0.687	0.045	
	work	0.745	0.579	0.166	
	liesure	0.711	0.672	0.039	
Football	list	0.681	0.77	-0.089	0.996
	text	0.621	0.679	-0.058	
	follows	0.556	0.672	-0.116	
	mentions	0.58	0.738	-0.158	
	retweets	0.634	0.752	-0.118	
Olympics	list	0.774	0.816	-0.042	0.996
	text	0.747	0.774	-0.027	
	follows	0.703	0.789	-0.086	
	mentions	0.71	0.889	-0.179	
	retweets	0.782	0.916	-0.134	

Politics Ire	list	0.591	0.625	-0.034	0.999
	text	0.445	0.552	-0.107	
	follows	0.491	0.552	-0.061	
	mentions	0.54	0.659	-0.119	
	retweets	0.6101	0.716	-0.1059	
Politics UK	list	0.585	0.591	-0.006	1
	text	0.491	0.516	-0.025	
	follows	0.478	0.524	-0.046	
	mentions	0.379	0.365	0.014	
	retweets	0.383	0.382	0.001	
Rugby	list	0.717	0.646	0.071	0.995
	text	0.722	0.65	0.072	
	follows	0.5903	0.591	-0.0007	
	mentions	0.593	0.617	-0.024	
	retweets	0.6296	0.655	-0.0254	

Table A.4: Direct Integration Modularity Clustering (Multiplicative) results for all of the multi-view, social-based, benchmark data sets. Performance results are detailed for each data set and each view of each data set.

Appendix B

Additional Results from COVID-19 Twitter Hashtag Clustering

This appendix contains additional results from the multi-view clustering of COIVD-19 Twitter hashtags. The first three tables display daily (and view, where relevant) results for the different aspects of multi-view clustering. These results include graph statistics on the graph learning for each of the views of the data, performance of the MVMC method, and clustering statistics for each of the daily clusterings. The final two tables in the appendix have additional hashtag examples from the clusterings investigated in Chapter four.

Date	View	Number of Nodes	Number of Edges	Density	Number of Weakly Connected Components	Number of Isolates
2/1	co_occurence	15287	1308045	0.011195	1050	881
	text	15287	1573200	0.013465	393	392
	URLs	15287	1318300	0.011283	4448	4447
	user	15287	439109	0.003758	326	273
2/2	co_occurence	16149	1438860	0.011035	1001	841
	text	16149	1695823	0.013006	471	470
	URLs	16149	1453646	0.011149	4565	4564
	user	16149	537180	0.00412	285	232
2/3	co_occurence	17982	1693603	0.010476	1151	972
	text	17982	2031992	0.012569	329	328
	URLs	17982	1783067	0.011029	4550	4549
	user	17982	573124	0.003545	351	290
2/4	co_occurence	16652	1508970	0.010884	1135	932
	text	16652	1818503	0.013117	346	345
	URLs	16652	1570702	0.01133	4364	4363
	user	16652	515832	0.003721	393	304
2/5	co_occurence	16152	1429575	0.01096	1076	915
	text	16152	1725481	0.013229	410	407
	URLs	16152	1504654	0.011536	4170	4169
	user	16152	465291	0.003567	381	306
2/6	co_occurence	15266	1303788	0.01119	1010	829
	text	15266	1575807	0.013524	401	400

	URLs	15266	1376797	0.011816	3955	3954
	user	15266	444828	0.003818	382	297
2/7	co_occurence	15374	1326558	0.011226	1015	855
	text	15374	1587152	0.013431	445	444
	URLs	15374	1414551	0.01197	3754	3753
	user	15374	466280	0.003946	341	276
2/8	co_occurence	13118	1049068	0.012194	789	651
	text	13118	1247921	0.014505	327	326
	URLs	13118	1090407	0.012674	3432	3431
	user	13118	408312	0.004746	271	213
2/9	co_occurence	12584	993081	0.012543	778	641
	text	12584	1167841	0.014751	350	349
	URLs	12584	1014517	0.012814	3403	3402
	user	12584	418691	0.005288	278	220
2/10	co_occurence	15179	1308448	0.011359	970	801
	text	15179	1581066	0.013725	340	339
	URLs	15179	1392744	0.01209	3738	3737
	user	15179	496210	0.004308	367	293
2/11	co_occurence	15239	1313626	0.011314	965	806
	text	15239	1587104	0.013669	343	342
	URLs	15239	1381755	0.011901	3888	3887
	user	15239	466958	0.004022	352	287
2/12	co_occurence	14177	1174430	0.011687	923	756
	text	14177	1436852	0.014299	307	306
	URLs	14177	1273819	0.012677	3360	3359
	user	14177	418024	0.00416	331	270
2/13	co_occurence	14386	1207856	0.011673	910	766
	text	14386	1436204	0.01388	387	386
	URLs	14386	1289969	0.012467	3414	3413
	user	14386	457879	0.004425	325	264
2/14	co_occurence	12295	959264	0.012692	789	655
	text	12295	1138638	0.015066	350	349
	URLs	12295	1038141	0.013736	2740	2739
	user	12295	371355	0.004914	298	236
2/15	co_occurence	10515	760052	0.01375	582	490
	text	10515	882256	0.015961	363	362
	URLs	10515	802917	0.014525	2526	2525
	user	10515	330037	0.005971	230	193
2/16	co_occurence	10115	721737	0.01411	574	471
	text	10115	838563	0.016394	257	256
	URLs	10115	757437	0.014808	2428	2427
	user	10115	341600	0.006678	214	164
2/17	co_occurence	12664	1014749	0.012656	712	590
	text	12664	1174771	0.014651	385	384
	URLs	12664	1091110	0.013608	2787	2786
	user	12664	388790	0.004849	277	216

2/18	co_occurence	12323	961434	0.012663	748	604
	text	12323	1146372	0.015099	336	335
	URLs	12323	1049730	0.013826	2735	2734
	user	12323	359539	0.004736	294	232
2/19	co_occurence	11919	918917	0.012938	711	572
	text	11919	1075561	0.015143	412	411
	URLs	11919	970881	0.01367	2891	2890
	user	11919	340275	0.004791	309	240
2/20	co_occurence	11767	896448	0.01295	627	499
	text	11767	1051676	0.015192	328	327
	URLs	11767	962209	0.0139	2722	2721
	user	11767	356822	0.005155	253	192
2/21	co_occurence	13975	1176267	0.012047	720	579
	text	13975	1384228	0.014176	374	373
	URLs	13975	1208756	0.012379	3605	3604
	user	13975	520516	0.005331	277	210
2/22	co_occurence	14848	1300698	0.0118	675	554
	text	14848	1476573	0.013396	508	507
	URLs	14848	1266354	0.011489	4236	4235
	user	14848	601471	0.005457	202	159
2/23	co_occurence	16093	1460843	0.011282	739	616
	text	16093	1697379	0.013109	493	491
	URLs	16093	1446184	0.011169	4464	4463
	user	16093	718200	0.005547	191	151
2/24	co_occurence	23657	2612548	0.009337	1270	1105
	text	23657	3077836	0.011	625	624
	URLs	23657	2643584	0.009448	6227	6226
	user	23657	1063342	0.0038	343	284
2/25	co_occurence	25485	2862600	0.008815	1666	1417
	text	25485	3507437	0.010801	494	493
	URLs	25485	2941166	0.009057	6846	6845
	user	25485	1072493	0.003303	398	316
2/26	co_occurence	27448	3184299	0.008454	1905	1630
	text	27448	3939629	0.010459	579	578
	URLs	27448	3289268	0.008732	7380	7379
	user	27448	1079997	0.002867	493	405
2/27	co_occurence	28844	3445061	0.008282	1935	1661
	text	28844	4262252	0.010246	614	612
	URLs	28844	3548242	0.00853	7716	7715
	user	28844	1084069	0.002606	524	426
2/28	co_occurence	28665	3433800	0.008358	1955	1666
	text	28665	4278387	0.010414	420	419
	URLs	28665	3517691	0.008562	7727	7726
	user	28665	1035504	0.002521	567	454
2/29	co_occurence	26679	3062861	0.008607	1737	1478

	text	26679	3801492	0.010682	555	554
	URLs	26679	3112844	0.008747	7440	7439
	user	26679	1024505	0.002879	414	330
3/1	co_occurence	26560	2990148	0.008478	1842	1545
	text	26560	3733658	0.010586	519	518
	URLs	26560	3116878	0.008837	7153	7152
	user	26560	1096898	0.00311	460	359
3/2	co_occurence	30791	3786996	0.007989	2141	1832
	text	30791	4717547	0.009952	584	583
	URLs	30791	4032933	0.008508	7603	7602
	user	30791	1189952	0.00251	587	469
3/3	co_occurence	33534	4343459	0.007725	2180	1843
	text	33534	5393075	0.009592	615	614
	URLs	33534	4531931	0.00806	8633	8632
	user	33534	1270716	0.00226	598	484
3/4	co_occurence	34298	4483468	0.007623	2300	1972
	text	34298	5551101	0.009438	636	635
	URLs	34298	4731239	0.008044	8581	8580
	user	34298	1296269	0.002204	650	531
3/5	co_occurence	36655	5002809	0.007447	2315	1988
	text	36655	6150833	0.009156	637	636
	URLs	36655	5225353	0.007778	9143	9142
	user	36655	1398143	0.002081	700	570
3/6	co_occurence	37478	5143087	0.007323	2415	2065
	text	37478	6343489	0.009033	677	676
	URLs	37478	5377217	0.007657	9459	9458
	user	37478	1395044	0.001986	680	568
3/7	co_occurence	32563	4183264	0.007891	1964	1682
	text	32563	5103512	0.009626	629	627
	URLs	32563	4219169	0.007958	8963	8962
	user	32563	1210568	0.002283	578	481
3/8	co_occurence	31165	3916897	0.008066	1831	1562
	text	31165	4783629	0.009851	639	638
	URLs	31165	3973506	0.008182	8424	8423
	user	31165	1308121	0.002694	472	383
3/9	co_occurence	34513	4595183	0.007716	2062	1752
	text	34513	5662158	0.009507	601	599
	URLs	34513	4788394	0.00804	8465	8464
	user	34513	1312631	0.002204	566	451
3/10	co_occurence	33710	4435573	0.007807	2237	1921
	text	33710	5512449	0.009702	497	496
	URLs	33710	4591942	0.008082	8489	8488
	user	33710	1112630	0.001958	630	516
3/11	co_occurence	29834	3710000	0.008337	1949	1684
	text	29834	4549186	0.010222	518	516
	URLs	29834	3801714	0.008543	7602	7601

	user	29834	850497	0.001911	577	476
3/12	co_occurence	32538	4232159	0.007995	2154	1881
	text	32538	5277652	0.00997	334	333
	URLs	32538	4174161	0.007886	9237	9236
	user	32538	789361	0.001491	680	563
3/13	co_occurence	34145	4554410	0.007813	2195	1892
	text	34145	5686434	0.009755	370	369
	URLs	34145	4511521	0.007739	9520	9519
	user	34145	788741	0.001353	740	615
3/14	co_occurence	36504	5116062	0.007679	2038	1781
	text	36504	6284669	0.009433	414	413
	URLs	36504	4774196	0.007166	11393	11392
	user	36504	951724	0.001428	593	484
3/15	co_occurence	40083	5906857	0.007353	1927	1665
	text	40083	7217267	0.008984	529	528
	URLs	40083	5339586	0.006647	13264	13263
	user	40083	1134177	0.001412	502	410
3/16	co_occurence	44579	6921264	0.006966	2294	2020
	text	44579	8550334	0.008605	494	493
	URLs	44579	6608853	0.006651	13144	13143
	user	44579	1174207	0.001182	678	579
3/17	co_occurence	46065	7217129	0.006802	2465	2164
	text	46065	8994908	0.008478	405	404
	URLs	46065	7059034	0.006653	12967	12966
	user	46065	1215076	0.001145	735	611
3/18	co_occurence	49377	8031118	0.006588	2571	2214
	text	49377	9999982	0.008203	499	498
	URLs	49377	7899346	0.00648	13671	13670
	user	49377	1342326	0.001101	760	629
3/19	co_occurence	50435	8327465	0.006548	2538	2200
	text	50435	10297722	0.008097	583	581
	URLs	50435	8137095	0.006398	13987	13986
	user	50435	1388644	0.001092	749	617
3/20	co_occurence	51776	8574626	0.006397	2777	2405
	text	51776	10766590	0.008033	509	508
	URLs	51776	8547938	0.006377	14006	14005
	user	51776	1424429	0.001063	805	669
3/21	co_occurence	49131	7880256	0.006529	2664	2315
	text	49131	9788678	0.008111	585	583
	URLs	49131	7569188	0.006272	14750	14749
	user	49131	1474112	0.001221	720	591
3/22	co_occurence	49396	8097011	0.006637	2236	1951
	text	49396	9866549	0.008088	705	704
	URLs	49396	7384271	0.006053	16014	16013
	user	49396	1580565	0.001296	539	442
3/23	co_occurence	56714	9973083	0.006201	2555	2267

	text	56714	12284376	0.007639	625	624
	URLs	56714	9565130	0.005948	16394	16393
	user	56714	1794422	0.001116	677	578
3/24	co_occurence	58548	10374631	0.006053	2786	2421
	text	58548	12834657	0.007489	733	732
	URLs	58548	10089279	0.005887	16553	16552
	user	58548	1799006	0.00105	724	599
3/25	co_occurence	60968	10959787	0.005897	3001	2582
	text	60968	13658304	0.007349	676	675
	URLs	60968	10871376	0.005849	16638	16637
	user	60968	1894732	0.001019	824	666
3/26	co_occurence	61735	11192358	0.005873	3114	2698
	text	61735	13912644	0.007301	733	731
	URLs	61735	11117281	0.005834	16772	16771
	user	61735	1928239	0.001012	851	695
3/27	co_occurence	61691	11177223	0.005874	3125	2712
	text	61691	13882971	0.007296	766	765
	URLs	61691	11096333	0.005831	16810	16809
	user	61691	1910421	0.001004	863	704
3/28	co_occurence	57009	9916477	0.006103	2813	2438
	text	57009	12166283	0.007487	850	849
	URLs	57009	9581732	0.005897	16594	16593
	user	57009	1918179	0.00118	731	595
3/29	co_occurence	54611	9168923	0.006149	3006	2589
	text	54611	11321969	0.007593	882	880
	URLs	54611	8998724	0.006035	15832	15831
	user	54611	1932976	0.001296	768	611
3/30	co_occurence	62664	11313175	0.005762	3291	2847
	text	62664	14150903	0.007208	897	896
	URLs	62664	11539559	0.005877	16355	16354
	user	62664	2143819	0.001092	913	752
3/31	co_occurence	66817	12564544	0.005629	3328	2892
	text	66817	15642964	0.007008	820	818
	URLs	66817	12613459	0.005651	17778	17777
	user	66817	2342428	0.001049	887	721
4/1	co_occurence	68636	13086119	0.005556	3326	2873
	text	68636	16218260	0.006886	916	915
	URLs	68636	12954633	0.0055	18851	18850
	user	68636	2411780	0.001024	864	701
4/2	co_occurence	68159	12921278	0.005563	3406	2922
	text	68159	16122006	0.006941	904	903
	URLs	68159	13000447	0.005597	18199	18198
	user	68159	2423759	0.001043	918	750
4/3	co_occurence	71068	13704636	0.005427	3571	3095
	text	71068	17133196	0.006785	848	847
	URLs	71068	13805058	0.005467	19014	19013

	user	71068	2593038	0.001027	969	788
4/4	co_occurence	64392	11692677	0.00564	3375	2870
	text	64392	14552274	0.007019	926	925
	URLs	64392	11635720	0.005613	18219	18218
	user	64392	2507174	0.001209	847	684
4/5	co_occurence	61647	10945327	0.00576	3295	2836
	text	61647	13544070	0.007128	1034	1032
	URLs	61647	10811578	0.00569	17864	17863
	user	61647	2431118	0.001279	843	681
4/6	co_occurence	67767	12628068	0.0055	3761	3221
	text	67767	15832847	0.006895	1008	1007
	URLs	67767	13184095	0.005742	16889	16888
	user	67767	2508564	0.001093	981	784
4/7	co_occurence	72653	14092171	0.00534	3826	3267
	text	72653	17657535	0.006691	909	906
	URLs	72653	14316697	0.005425	19273	19272
	user	72653	2689533	0.001019	1054	847
4/8	co_occurence	75832	15043486	0.005232	3840	3290
	text	75832	18832764	0.00655	1046	1045
	URLs	75832	15401513	0.005357	19660	19659
	user	75832	2957848	0.001029	1087	876
4/9	co_occurence	75722	15026145	0.005241	3872	3268
	text	75722	18803070	0.006559	1016	1015
	URLs	75722	15302849	0.005338	19898	19897
	user	75722	2982852	0.00104	1072	817
4/10	co_occurence	72566	14099819	0.005355	3709	3192
	text	72566	17505343	0.006649	1077	1075
	URLs	72566	14183413	0.005387	19658	19657
	user	72566	2912466	0.001106	921	741
4/11	co_occurence	66206	12162470	0.00555	3567	3009
	text	66206	15125876	0.006902	1021	1019
	URLs	66206	12081651	0.005513	19010	19009
	user	66206	2717480	0.00124	926	714
4/12	co_occurence	65675	12085740	0.005604	3465	2935
	text	65675	14790467	0.006858	1214	1213
	URLs	65675	11788072	0.005466	19253	19252
	user	65675	2896898	0.001343	858	685
4/13	co_occurence	72887	14052141	0.00529	3866	3306
	text	72887	17482253	0.006582	1125	1124
	URLs	72887	14355911	0.005405	19171	19170
	user	72887	3144790	0.001184	1046	819
4/14	co_occurence	76917	15339579	0.005186	4228	3610
	text	76917	19141688	0.006471	1133	1131
	URLs	76917	15889766	0.005372	19213	19212
	user	76917	2980272	0.001008	1237	997
4/15	co_occurence	77407	15499784	0.005174	4148	3553

	text	77407	19394901	0.006474	1211	1210
	URLs	77407	16001360	0.005341	19530	19529
	user	77407	2981994	0.000995	1128	917
4/16	co_occurence	79653	16148152	0.00509	4227	3584
	text	79653	20299610	0.006399	1145	1144
	URLs	79653	16661657	0.005252	20291	20290
	user	79653	3106543	0.000979	1178	929
4/17	co_occurence	78367	15592973	0.005078	4491	3772
	text	78367	19733196	0.006426	1106	1105
	URLs	78367	16331688	0.005319	19633	19632
	user	78367	3098978	0.001009	1225	956
4/18	co_occurence	67899	12440165	0.005397	4058	3423
	text	67899	15659217	0.006793	1248	1247
	URLs	67899	12885653	0.00559	18124	18123
	user	67899	2705727	0.001174	1096	868
4/19	co_occurence	61885	10815632	0.005648	3372	2842
	text	61885	13507919	0.007054	1237	1236
	URLs	61885	11030604	0.005761	17193	17192
	user	61885	2579525	0.001347	906	711
4/20	co_occurence	79264	15968835	0.005083	4141	3540
	text	79264	19961976	0.006355	1344	1343
	URLs	79264	16492068	0.00525	20379	20378
	user	79264	3347808	0.001066	1157	938
4/21	co_occurence	80765	16467424	0.005049	4216	3580
	text	80765	20666618	0.006337	1252	1250
	URLs	80765	16959057	0.0052	20862	20861
	user	80765	3291650	0.001009	1155	909
4/22	co_occurence	81697	16734768	0.005015	4325	3682
	text	81697	21036452	0.006304	1188	1187
	URLs	81697	17245081	0.005168	21000	20999
	user	81697	3334385	0.000999	1215	972
4/23	co_occurence	83041	17029111	0.004939	4453	3796
	text	83041	21518276	0.006241	1298	1296
	URLs	83041	17803253	0.005164	21028	21027
	user	83041	3535057	0.001025	1300	1027
4/24	co_occurence	76507	15010702	0.005129	4442	3800
	text	76507	19043433	0.006507	1100	1099
	URLs	76507	15794897	0.005397	19086	19085
	user	76507	2996896	0.001024	1311	1048
4/25	co_occurence	72156	13606908	0.005227	4296	3618
	text	72156	17036558	0.006544	1308	1307
	URLs	72156	14148577	0.005435	19140	19139
	user	72156	3135195	0.001204	1173	907
4/26	co_occurence	70778	13131963	0.005243	4243	3538
	text	70778	16482923	0.006581	1435	1431
	URLs	70778	13581716	0.005422	19486	19485

	user	70778	3217555	0.001285	1107	843
4/27	co_occurence	83252	17028252	0.004914	4488	3766
	text	83252	21411786	0.006179	1502	1501
	URLs	83252	17940021	0.005177	20733	20732
	user	83252	3699063	0.001067	1346	1069
4/28	co_occurence	84755	17561109	0.004889	4550	3841
	text	84755	22145918	0.006166	1315	1313
	URLs	84755	18331847	0.005104	21554	21553
	user	84755	3664427	0.00102	1257	982
4/29	co_occurence	86072	18018373	0.004864	4585	3879
	text	86072	22691930	0.006126	1246	1245
	URLs	86072	18728588	0.005056	21944	21943
	user	86072	3674244	0.000992	1302	1041
4/30	co_occurence	88539	18634524	0.004754	4878	4090
	text	88539	23644676	0.006033	1395	1393
	URLs	88539	19629344	0.005008	22228	22227
	user	88539	3803211	0.00097	1428	1136

Table B.1: Graph metrics for each of the view graphs learned by a symmetric kNN procedure for each view for each of the days.

Date	View	Modularity	Resolution	Weights	Iterations
2/1	co_occurence	0.278001	4.380331	1.482102	7
	text	0.16435	2.95081	1.087881	
	URLs	0.005516	1.162796	0.147665	
	user	0.173467	3.930365	1.282352	
2/2	co_occurence	0.286614	3.938136	1.441558	6
	text	0.188447	3.006953	1.143299	
	URLs	0.00555	1.145516	0.135136	
	user	0.192949	3.683846	1.280007	
2/3	co_occurence	0.286241	3.942712	1.457338	6
	text	0.192167	2.970779	1.159595	
	URLs	0.004644	1.118974	0.114302	
	user	0.187434	3.567321	1.268766	
2/4	co_occurence	0.270026	4.041021	1.491818	6
	text	0.172387	2.872412	1.136352	
	URLs	0.003812	1.096782	0.095418	
	user	0.171393	3.60256	1.276411	
2/5	co_occurence	0.284235	4.26236	1.525097	6
	text	0.149828	2.738225	1.044148	
	URLs	0.005719	1.169057	0.157053	
	user	0.17069	3.721612	1.273701	
2/6	co_occurence	0.288712	4.211175	1.558165	6
	text	0.142493	2.731876	1.062701	
	URLs	0.005725	1.160536	0.15406	
	user	0.14223	3.483169	1.225075	

2/7	co_occurence	0.283714	4.291961	1.538853	6
	text	0.14339	2.878466	1.085718	
	URLs	0.004994	1.163499	0.152936	
	user	0.140694	3.546961	1.222493	
2/8	co_occurence	0.280486	4.33098	1.519178	6
	text	0.144824	2.896395	1.067268	
	URLs	0.006659	1.194224	0.176815	
	user	0.14431	3.673437	1.236739	
2/9	co_occurence	0.255646	4.601571	1.489243	6
	text	0.143518	3.156272	1.107901	
	URLs	0.006713	1.215803	0.189922	
	user	0.135231	3.767356	1.212934	
2/10	co_occurence	0.283925	4.235479	1.533476	6
	text	0.145416	2.890653	1.088735	
	URLs	0.005304	1.149715	0.14289	
	user	0.146814	3.574949	1.2349	
2/11	co_occurence	0.28832	4.179579	1.552373	6
	text	0.141737	2.726751	1.054215	
	URLs	0.004601	1.13858	0.134586	
	user	0.152609	3.589321	1.258826	
2/12	co_occurence	0.289219	4.507069	1.564102	7
	text	0.121123	2.81027	1.026068	
	URLs	0.004084	1.161964	0.150906	
	user	0.133947	3.798863	1.258924	
2/13	co_occurence	0.284712	4.399588	1.538355	6
	text	0.124933	2.858436	1.044586	
	URLs	0.006907	1.22792	0.203677	
	user	0.136964	3.619306	1.213382	
2/14	co_occurence	0.259428	4.480011	1.508276	6
	text	0.127484	2.933564	1.054594	
	URLs	0.006106	1.210207	0.188383	
	user	0.137655	3.861762	1.248746	
2/15	co_occurence	0.267328	4.645225	1.490221	6
	text	0.128597	2.943381	1.014916	
	URLs	0.008547	1.263863	0.221435	
	user	0.164418	4.098882	1.273428	
2/16	co_occurence	0.267123	4.738217	1.515482	6
	text	0.145081	3.197504	1.117502	
	URLs	0.003194	1.088496	0.08436	
	user	0.160704	4.007889	1.282657	
2/17	co_occurence	0.269745	4.288869	1.467299	6
	text	0.142595	2.989108	1.077175	
	URLs	0.007121	1.20467	0.183143	
	user	0.161948	3.939208	1.272383	
2/18	co_occurence	0.269736	4.710909	1.472488	7

	text	0.141948	3.058842	1.044374	
	URLs	0.007816	1.26908	0.220965	
	user	0.170013	4.139215	1.262173	
2/19	co_occurence	0.284727	4.17269	1.517432	6
	text	0.126586	2.704225	1.001628	
	URLs	0.007945	1.249613	0.221538	
	user	0.159276	3.761693	1.259402	
2/20	co_occurence	0.266664	4.556653	1.403855	7
	text	0.153198	3.108558	1.036448	
	URLs	0.008953	1.344939	0.26308	
	user	0.178017	4.510735	1.296617	
2/21	co_occurence	0.286969	3.61909	1.37189	6
	text	0.226821	3.049316	1.19249	
	URLs	0.007058	1.195812	0.175302	
	user	0.256497	3.194589	1.260318	
2/22	co_occurence	0.297618	1.324339	1.133198	2
	text	0.46965	1.197193	1.514987	
	URLs	0.008295	1.020888	0.0576	
	user	0.374242	1.212409	1.294215	
2/23	co_occurence	0.248923	1.271907	1.039907	2
	text	0.519247	1.12657	1.713544	
	URLs	9.27E-05	1.015685	0.04949	
	user	0.365007	1.186161	1.197059	
2/24	co_occurence	0.222656	1.27289	0.916761	2
	text	0.570381	1.115851	1.828699	
	URLs	-0.00575	1.010895	0.032384	
	user	0.355543	1.227456	1.222156	
2/25	co_occurence	0.20764	1.323024	0.986665	2
	text	0.563256	1.110726	1.774222	
	URLs	-0.00382	1.009106	0.026686	
	user	0.359003	1.244989	1.212427	
2/26	co_occurence	0.192079	1.270485	0.936145	2
	text	0.599918	1.039578	1.842637	
	URLs	-0.00033	1.006518	0.020718	
	user	0.360398	1.212687	1.200501	
2/27	co_occurence	0.294948	3.366221	1.472908	7
	text	0.240941	2.604637	1.239237	
	URLs	0.000897	1.01579	0.018122	
	user	0.233629	2.782861	1.269732	
2/28	co_occurence	0.291972	3.337084	1.460744	7
	text	0.229218	2.608077	1.219073	
	URLs	0.002165	1.029746	0.033401	
	user	0.234154	2.874624	1.286781	
2/29	co_occurence	0.255563	1.415784	1.088322	2
	text	0.446156	1.380516	1.585263	
	URLs	-0.00764	1.01466	0.035537	

	user	0.359947	1.401837	1.290878	
3/1	co_occurence	0.199918	1.287432	0.978643	2
	text	0.525888	1.111482	1.712439	
	URLs	-0.00072	1.014175	0.041277	
	user	0.35313	1.239646	1.267641	
3/2	co_occurence	0.164966	1.22652	0.971942	2
	text	0.571249	1.016744	1.83445	
	URLs	-0.00245	1.006956	0.025928	
	user	0.323212	1.12456	1.16768	
3/3	co_occurence	0.173916	1.200179	0.954417	2
	text	0.567749	0.991141	1.870842	
	URLs	-0.00313	1.005763	0.022916	
	user	0.325724	1.108868	1.151825	
3/4	co_occurence	0.306137	3.743453	2.781681	8
	text	0.192494	2.633356	2.055932	
	URLs	-0.08256	0.379029	-3.01984	
	user	0.188126	2.889708	2.182231	
3/5	co_occurence	0.292791	3.563284	1.459546	7
	text	0.216334	2.852342	1.231618	
	URLs	0.002351	1.046183	0.04893	
	user	0.216238	3.005523	1.259907	
3/6	co_occurence	0.183164	1.256289	0.996827	2
	text	0.533682	1.041736	1.758472	
	URLs	-0.00221	1.006483	0.023674	
	user	0.346764	1.1589	1.221027	
3/7	co_occurence	0.296231	3.625903	1.484642	6
	text	0.216304	2.846983	1.232028	
	URLs	0.001625	1.030499	0.032776	
	user	0.229335	2.871041	1.250553	
3/8	co_occurence	0.224349	1.248575	1.049332	2
	text	0.517919	1.05657	1.660098	
	URLs	0.000763	1.008385	0.028149	
	user	0.36423	1.179232	1.26242	
3/9	co_occurence	0.22108	1.305327	1.000826	2
	text	0.486818	1.223353	1.663571	
	URLs	0.002005	1.012658	0.032226	
	user	0.351968	1.321711	1.303378	
3/10	co_occurence	0.186207	1.236602	0.998943	2
	text	0.571919	0.999572	1.810448	
	URLs	-0.00098	1.006107	0.022279	
	user	0.355327	1.122833	1.16833	
3/11	co_occurence	0.225269	1.364663	1.172956	2
	text	0.510299	1.122007	1.685447	
	URLs	0.000979	1.007982	0.025093	
	user	0.347578	1.172693	1.116503	
3/12	co_occurence	0.290565	1.349464	1.294983	2

	text	0.480265	1.088079	1.628612	
	URLs	0.002657	1.006759	0.022509	
	user	0.294533	1.181421	1.053896	
3/13	co_occurence	0.346058	3.883607	1.623038	7
	text	0.183697	2.525161	1.086668	
	URLs	0.001452	1.031608	0.034073	
	user	0.19911	3.116072	1.256221	
3/14	co_occurence	0.260563	1.306147	1.199402	2
	text	0.543815	1.047454	1.694076	
	URLs	0.001478	1.005692	0.019252	
	user	0.329737	1.128263	1.08727	
3/15	co_occurence	0.296067	1.312708	1.17553	2
	text	0.548048	1.044337	1.673799	
	URLs	0.000827	1.005254	0.016748	
	user	0.357103	1.139128	1.133922	
3/16	co_occurence	0.289601	1.244605	1.097948	2
	text	0.555524	0.999444	1.76673	
	URLs	-0.00028	1.004935	0.017222	
	user	0.34675	1.120048	1.1181	
3/17	co_occurence	0.382162	3.981867	1.681548	7
	text	0.179203	2.516691	1.06546	
	URLs	0.001044	1.01111	0.012069	
	user	0.216385	3.004113	1.240923	
3/18	co_occurence	0.277646	1.281594	1.133369	2
	text	0.542662	1.026866	1.733403	
	URLs	-0.00227	1.00477	0.016042	
	user	0.339402	1.150873	1.117186	
3/19	co_occurence	0.281085	1.23744	1.124222	2
	text	0.554111	0.999218	1.727616	
	URLs	-0.00503	1.003669	0.013221	
	user	0.362303	1.118883	1.134941	
3/20	co_occurence	0.294435	1.32065	1.193985	2
	text	0.514237	1.069149	1.688807	
	URLs	-0.00203	1.004821	0.01542	
	user	0.351251	1.164605	1.101788	
3/21	co_occurence	0.31872	1.325723	1.129282	2
	text	0.517221	1.080985	1.670099	
	URLs	-0.00211	1.005442	0.016148	
	user	0.351594	1.214646	1.184472	
3/22	co_occurence	0.345428	1.380258	1.195941	2
	text	0.495224	1.124381	1.616446	
	URLs	-0.00502	1.005613	0.016106	
	user	0.353357	1.231984	1.171508	
3/23	co_occurence	0.372539	4.303172	1.638168	7
	text	0.186882	2.794489	1.10104	
	URLs	0.000538	0.994394	-0.00581	

	user	0.219748	3.352638	1.266605	
3/24	co_occurence	0.319018	1.302235	1.205148	2
	text	0.513187	1.064847	1.662284	
	URLs	-0.00393	1.004702	0.014826	
	user	0.337893	1.175806	1.117742	
3/25	co_occurence	0.379585	4.067591	1.650353	7
	text	0.184089	2.65299	1.084851	
	URLs	7.37E-05	1.00744	0.007875	
	user	0.225739	3.125658	1.256921	
3/26	co_occurence	0.390823	3.849522	1.642042	6
	text	0.1845	2.611706	1.085492	
	URLs	0.001215	1.00162	0.001738	
	user	0.230393	3.137124	1.270728	
3/27	co_occurence	0.33	1.306002	1.235194	2
	text	0.478559	1.073015	1.651149	
	URLs	-0.00819	1.00465	0.015555	
	user	0.313303	1.158017	1.098102	
3/28	co_occurence	0.359416	3.956025	1.583222	6
	text	0.186968	2.739051	1.100696	
	URLs	0.000595	1.009867	0.010345	
	user	0.235748	3.359735	1.305737	
3/29	co_occurence	0.331773	1.313231	1.19494	2
	text	0.473517	1.101318	1.646497	
	URLs	-0.00543	1.005035	0.016006	
	user	0.323544	1.175837	1.142557	
3/30	co_occurence	0.341181	1.311435	1.242418	2
	text	0.469658	1.085219	1.640168	
	URLs	-0.00882	1.00478	0.015348	
	user	0.316228	1.14502	1.102065	
3/31	co_occurence	0.345672	1.305938	1.221588	2
	text	0.48013	1.072957	1.657809	
	URLs	-0.00256	1.005277	0.016536	
	user	0.319679	1.140275	1.104067	
4/1	co_occurence	0.373617	3.905897	1.610216	6
	text	0.194402	2.709749	1.111916	
	URLs	0.000804	0.996072	-0.00419	
	user	0.232529	3.246359	1.282054	
4/2	co_occurence	0.343492	1.326723	1.240169	2
	text	0.475614	1.08156	1.632088	
	URLs	-0.00372	1.004315	0.013805	
	user	0.321531	1.152127	1.113938	
4/3	co_occurence	0.291739	1.284101	1.178471	2
	text	0.501056	1.055677	1.66636	
	URLs	-0.00622	1.004488	0.014926	
	user	0.356098	1.137683	1.140243	
4/4	co_occurence	0.368742	4.05489	1.59219	6

	text	0.183	2.727437	1.076657	
	URLs	0.001294	1.033441	0.033557	
	user	0.243818	3.39333	1.297597	
4/5	co_occurence	0.361219	4.192373	1.608245	6
	text	0.188753	2.819537	1.114154	
	URLs	-0.00046	0.973585	-0.02759	
	user	0.234523	3.475448	1.305191	
4/6	co_occurence	0.366153	3.729305	1.562138	6
	text	0.190957	2.701268	1.119403	
	URLs	0.000518	0.99387	-0.00657	
	user	0.249396	3.283236	1.325024	
4/7	co_occurence	0.346924	4.263688	3.119555	8
	text	0.178003	2.772574	2.119908	
	URLs	-0.07697	0.333095	-3.64819	
	user	0.196012	3.299402	2.408727	
4/8	co_occurence	0.317079	1.301076	1.237214	2
	text	0.48746	1.075788	1.663097	
	URLs	-0.00283	1.004938	0.015772	
	user	0.337741	1.141115	1.083917	
4/9	co_occurence	0.358883	1.310046	1.239854	2
	text	0.468538	1.082505	1.620417	
	URLs	-0.00851	1.004955	0.015625	
	user	0.320859	1.14754	1.124104	
4/10	co_occurence	0.362408	4.379696	3.126548	8
	text	0.169023	2.80675	2.060261	
	URLs	-0.07708	0.338058	-3.51399	
	user	0.206326	3.191934	2.327177	
4/11	co_occurence	0.382534	4.173565	1.620814	6
	text	0.17995	2.739352	1.068201	
	URLs	0.000759	1.008563	0.008691	
	user	0.245946	3.480686	1.302294	
4/12	co_occurence	0.374879	4.092456	1.534506	6
	text	0.190368	2.769957	1.048161	
	URLs	0.005116	1.120881	0.11035	
	user	0.260993	3.614613	1.306983	
4/13	co_occurence	0.372679	3.820528	1.542172	6
	text	0.187131	2.612595	1.044981	
	URLs	0.004505	1.092113	0.089151	
	user	0.271689	3.429403	1.323696	
4/14	co_occurence	0.378291	3.783146	1.566736	6
	text	0.189152	2.665697	1.071477	
	URLs	0.0031	1.05573	0.055897	
	user	0.253812	3.325769	1.30589	
4/15	co_occurence	0.370377	3.562185	1.560853	6
	text	0.191588	2.59113	1.079845	
	URLs	0.002432	1.034464	0.036178	

	user	0.262691	3.213574	1.323124	
4/16	co_occurence	0.370728	3.553086	1.595097	6
	text	0.195899	2.566821	1.117044	
	URLs	-0.00062	0.971799	-0.03194	
	user	0.2497	3.103623	1.319799	
4/17	co_occurence	0.357899	3.953116	3.216401	7
	text	0.165753	2.579116	2.095247	
	URLs	-0.07776	0.336657	-3.802	
	user	0.209556	3.130921	2.490352	
4/18	co_occurence	0.365308	4.39719	3.1863	8
	text	0.151244	2.680479	1.989463	
	URLs	-0.07838	0.345162	-3.48982	
	user	0.190545	3.166194	2.314056	
4/19	co_occurence	0.377645	4.25813	1.632867	6
	text	0.175616	2.782654	1.078959	
	URLs	-0.00031	1.007908	0.00804	
	user	0.218089	3.474154	1.280134	
4/20	co_occurence	0.373325	3.975643	1.617862	6
	text	0.179433	2.689335	1.076143	
	URLs	0.000493	0.997386	-0.00275	
	user	0.245725	3.382669	1.308741	
4/21	co_occurence	0.342113	1.253127	1.207957	2
	text	0.479471	1.061709	1.665021	
	URLs	-0.00536	1.004223	0.014363	
	user	0.317408	1.117807	1.11266	
4/22	co_occurence	0.381052	3.908378	1.647919	6
	text	0.174562	2.597945	1.065182	
	URLs	0.000623	0.996246	-0.00402	
	user	0.228575	3.257679	1.29092	
4/23	co_occurence	0.383648	3.721941	1.605334	6
	text	0.177987	2.589291	1.066264	
	URLs	0.001181	1.009656	0.010224	
	user	0.248143	3.275832	1.318178	
4/24	co_occurence	0.375464	3.695858	1.620669	6
	text	0.177568	2.550345	1.074038	
	URLs	0.000483	0.996654	-0.00364	
	user	0.243392	3.167788	1.308932	
4/25	co_occurence	0.381492	3.924147	1.630713	6
	text	0.168047	2.605728	1.053918	
	URLs	0.0013	1.012214	0.012799	
	user	0.224597	3.380724	1.302571	
4/26	co_occurence	0.378216	4.118803	1.61931	6
	text	0.168512	2.678507	1.050519	
	URLs	0.001409	1.030307	0.030516	
	user	0.232398	3.469266	1.299654	
4/27	co_occurence	0.381412	3.662215	1.600407	5

	text	0.179316	2.565846	1.064348	
	URLs	0.001499	1.009718	0.01036	
	user	0.258497	3.258019	1.324885	
4/28	co_occurence	0.368548	3.620964	1.592892	6
	text	0.184631	2.597744	1.093643	
	URLs	-0.00012	0.986469	-0.01481	
	user	0.241676	3.28754	1.328278	
4/29	co_occurence	0.346739	1.289088	1.232358	2
	text	0.45652	1.076153	1.659839	
	URLs	-0.00473	1.004425	0.014965	
	user	0.300365	1.158802	1.092838	
4/30	co_occurence	0.299287	1.2675	1.20931	2
	text	0.477057	1.068183	1.674795	
	URLs	-0.00727	1.00444	0.015318	
	user	0.333816	1.136924	1.100577	

Table B.2: Multi-view Modularity Clustering (MVMC) performance results for each view of each days' data. Generally, the weight of a view correlates to the important of that view in finding clusters by MVMC and the number of iterations correlates to the strength of the cluster structure present in the data.

Date	Number of Clusters	Filtered Number of Clusters (cluster size >5)	Mean of Filtered Clusters Sizes	STD of Filtered Clusters Sizes
2/1	119	76	180.6053	167.4215
2/2	109	69	209.8261	201.1601
2/3	103	70	230	223.8302
2/4	109	70	211.9857	188
2/5	119	80	180.8375	178.3021
2/6	119	70	194.5	167.7832
2/7	112	80	173.075	163.6346
2/8	111	77	153.8701	138.3024
2/9	132	90	126.8667	120.3382
2/10	113	79	173.4937	169.7875
2/11	120	74	185	178.7091
2/12	116	76	168.7632	151.3474
2/13	115	78	168.141	161.5889
2/14	115	79	140.5823	126.7781
2/15	113	81	118.1235	104.2561
2/16	118	86	107.0233	103.8421
2/17	113	80	143.25	137.5473
2/18	120	83	134.506	122.7264
2/19	102	73	147.1781	135.223
2/20	118	90	119.0778	125.4855
2/21	86	63	199.6508	216.4846
2/22	16	11	981.9091	691.8503
2/23	15	9	1153.111	870.5808
2/24	19	9	1654.556	1292.167

2/25	27	9	1717.778	1438.459
2/26	18	9	1737.444	1709.251
2/27	81	49	515.1224	507.0892
2/28	105	56	444.6429	475.7289
2/29	25	12	1770.167	1444.033
3/1	21	9	2133.333	1632.524
3/2	28	8	2352.625	1508.184
3/3	21	9	1994	1765.442
3/4	97	58	520.8276	543.1009
3/5	122	62	524.3387	579.0867
3/6	24	9	2701.667	1722.963
3/7	117	63	459.1905	508.5882
3/8	25	10	2018.8	1385.495
3/9	29	10	2478.5	1629.758
3/10	26	8	2561.75	1842.316
3/11	27	11	1826.455	1621.901
3/12	25	13	1679.923	1687.235
3/13	96	55	543.4182	451.5695
3/14	23	11	1906.182	2253.779
3/15	33	11	2171.636	2289.916
3/16	30	10	2483.4	2605.292
3/17	104	59	681.0678	569.2046
3/18	31	12	2404.083	2643.212
3/19	26	12	2411.167	2655.622
3/20	40	12	2924.5	2921.176
3/21	30	12	2593.917	2535.789
3/22	26	14	2296.143	2354.217
3/23	126	71	707.2535	654.7393
3/24	35	12	2999.333	2974.958
3/25	136	65	823.9077	756.9975
3/26	133	60	907.7667	799.3329
3/27	30	13	2994	2887.427
3/28	138	79	639.2025	719.539
3/29	35	15	2337.2	2362.898
3/30	36	12	3282.75	2847.888
3/31	32	12	3468.75	3025.41
4/1	134	66	914.5303	824.623
4/2	38	13	3308.692	3209.444
4/3	28	13	3503.077	3390.986
4/4	127	74	769.7838	728.8596
4/5	150	76	718.6842	695.6831
4/6	148	67	892.6269	895.985
4/7	134	74	869.3514	859.1948
4/8	35	13	3935.154	3798.074
4/9	36	12	3966.167	3332.758
4/10	150	75	861.96	861.9259

4/11	136	76	771.8684	756.2456
4/12	122	67	869.3881	784.6204
4/13	128	66	971.4394	896.3212
4/14	141	70	971.4	1020.046
4/15	137	61	1115.115	1126.078
4/16	140	66	1059.303	1138.024
4/17	119	71	977.3099	1021.644
4/18	137	77	788.7662	778.0003
4/19	148	81	681.9753	716.2485
4/20	156	74	948.5135	983.3239
4/21	34	13	3793.462	4157.574
4/22	156	64	1126.406	1067.546
4/23	150	65	1131.477	1111.553
4/24	134	67	1005.925	1020.836
4/25	129	70	911.1714	890.3115
4/26	115	69	909.7536	816.8968
4/27	129	66	1114.167	1085.461
4/28	157	67	1114.403	1140.764
4/29	43	14	3835.214	3988.021
4/30	44	14	4187.643	4338.355

Table B.3: Cluster statistics for each days' multi-view clusterings. Generally, clusters of size less than 5 hashtags typically contained erroneous hashtags, and so these clusters were excluded.

Period 1, Chinese Cluster	Period 1, Syrian Civil War Cluster	Period 2, Online Education Cluster
HongKongProstests	Assad_Torture	study
ChinaCommunistParty	TBT	online
AnimalRights	WORLD	DigitalLearning
standwithhongkong	Event201	edchat
SCMP	CANADA	edtech
expats	AssadCrimes	history
HubeiProvince	GOD	Education
seafood	PANDEMIC	child
BatSoup	BIOWEAPON	highereducation
Shangai	CORONAVIRUSWORLD	college
wuhanquaranted	InfoWars	kids
ChenQuishi	MEXICO	teaching
ZeroHedge	SINGAPORE	AcademicChatter
cononavirus	CHINESE	school
FreeTibet	Nation	highered
TaiwanCanHelp	Infectados	eLearning
AEWDynamite	WhiteHelmets	EdChat
WuhanCoronovirus	OUTBREAK	Teachers
DrLiWenliang	CORONA	intled
BoycottChina	CORONAVIRUS	AcademicTwitter
carrielam	AlexJonesShow	STEM

Pets	Caribbean	students
HKexit	FAKENEWS	universities
terrorist	BILLGATES	international
Macau	BEIJING	MedEd
Killing	Chemical_Assad	student
china_is_territorist	PANDEMIA	education
Hongkong	LGBT	math
SwineFever	RETWEEET	Learning
Muslims	jesus	elearning
JD	CINA	virtuallearning
Whistleblower	RETWEET	onlinelearning
MoonJaein	NCOV19	university
Dictatorship	Syrie	HigherEd
BioWeaponsExpert	fda	distancelearning
hkprotest	ISRAEL	STEMeducation
BeltandRoad	AssadGenocide	schools
ChineseCommunistParty	TRUMP	
hongkongpolice	VIRUSCORONA	
Virology	ALERT	
Tibetans	BBI	
Vietnamese	VIRUS	
HongKongers	BREAKINGNEWS	
Beating	CHINA	
trust	ACTUALIZACION	
5DemandsNot1Less	CONSPIRACY	
StopBeijing2022	globalists	
chinazi	HUBEI	
hkpolicebrutality	WUHAN	
myedgeprop	TheResistance1776	
CovID19	EBOLA	
DrLi	QUARANTINE	
Hubie	IndianArmy	
China_is_terrorists	HEALTH	
dogmeat	PutinAtWar	
China_is_terrorist	SaveIdlib	
Tibet	LAMORGESE	
CoronaCartoons	WW3	
WuhanStrong	Vimeo	
martiallaw	CORONAVIRUSOUTBREAK	
Communist	IdlibUnderFire	
Weibo	WUFLU	
NeverForget		
dangerous		
Coronarivus		
WuhanVirus		
HK		

HKProtests		
CloseTheBorder		
XiaoZhan		
Yulin		
SaveUyghurs		
conronavirus		
Shanghai		
HKgov		
AnimalWelfare		
NewsAlert		
antiELAB		
wuhanconoravirus		
ChinaDaily		
TsingYi		
BRI		
Wuhanhospital		
Rats		
CommunismKills		
Mao		

Table B.4: Additional example hashtags for each of the clusters of interest identified in Chapter Four. Note that these hashtags, as presented in this table, are not ordered in any way relative to their usage.

Period 1, U.S. Politics Cluster	Period 3, U.S. Politics
BernieSanders2020	BarackObama
Democrats	BernieSanders2020
Epstein	Democrats
ImpeachmentHearings	drudge
WETHEPEOPLE	FlatEarth
UnitedNations	Flu
xenophobia	Epstein
BlackLivesMatter	RevolutionNow
WakeUp	DumpTrump2020
GatesFoundation	CabalTakedown
WashingtonPost	CoronaVirusTruth
DumpTrump	ScienceMatters
USPolitics	WednesdayWisdom
Exclusive	donaldtrump
KAGA2020	TheRealDonaldTrump
BioWeapon	TruthMatters
MAGA	Infowars
Trump	GeorgiaGuidestones
Bernie2020	FDA
AmericaFirst	BlackLivesMatter
GOPComplicitTraitors	WakeUp
DrainingTheSwamp	Event201

IowaCaucusDisaster	AmericansFirst
YellowVests	GatesFoundation
immigration	DumpTrump
RIPAmerica	KAGA2020
StopTheMadness	BernieBeatsTrump
QAnon2020	socialism
VoteBlueToEndThisNightmare	MAGA
TrumpBudget	bioweapon
billgateseugenicist	vaccination
NeverAgainIsNow	Presidementia
KAG2020LandslideVictory	Whistleblower
Bats	CoronavirusUSA
SayNoToRacism	AmericaFirst
WashingtonDC	GOPComplicitTraitors
TheMoreYouKnow	alternativefacts
Qanons	tcot
imnotavirus	ElonMusk
Schiff	bigpharma
2A	conspiracytheories
TheGreatAwakeningWorldwide	CoronavirusVaccine
FoxNews	Woke
WakeUpAmerica	TrumpIsANationalSecurityThreat
VoteBlueNoMatterWho2020	StopTheMadness
Trump2020	NeverTrumper
ScottyfromMarketing	Chinatown
Illuminati	Breitbart
Agenda2030	JeffBezos
JoeBiden	QAnon2020
VoteBlue2020	PutinsGOP
IoNonSonoUnVirus	VaccinesKill
TrumpNotFitForOffice	Schumer
PatriotsFight	maga2020
PayAttention	KAG2020LandslideVictory
QArmy	SenatorForSale
Trump2020Landslide	Qanons
TrumpAdministration	Cult45
PatriotsUnite	Schiff
Trump2020LandslideVictory	HHS
TrumpsGuilty	FoxNews
NancyPelosi	WakeUpAmerica
Vaccine	CrookedHillary
bluelivesmatter	VoteBlueNoMatterWho2020
CloserNation	Trump2020
qanon	Depopulation
CoVID19	JoeBiden
q	VoteBlue2020

AlexJones	QArmy
nytimes	Trump2020Landslide
realDonaldTrump	TeaParty
AlternativeMedicine	TrumpAdministration
republicans	antivax
racist	illuminati
hillaryemails	Freemasons
billgates	TrumpResign
ImpeachTrump	Conspiracy
IdiotInChief	Fascism
stopracism	GOPCowards
RemoveTrump	TrumpsGuilty
IngrahamAngle	5g
EndTheFed	Covfefe
Hannity	qanon
BernieBros	RushLimbaugh
highered	AlexJones
Biowarfare	FAKENEWS
NRA	billgates
Globalists	NRA
Obama	chemtrails
conspiracy	Vaccines
chemtrails	CoronavirusCoverup
FakeNews	WWG1WGAWORLDWIDE
MakeAmericaGreatAgain	FalseFlag
QArmyJapanFlynn	ChineseCoronavirus
VoteBlueNoMatterWho	populationcontrol
AbolishICE	StopTheCoup
Biden	DeepState
Anons	DrainTheDeepState
TCOT	Zuckerberg
DemocratsTheEnemyWithin	WhiteSupremacy
woke	
DrainTheDeepState	
PatriotsAwakened	
CrimeMinisterMorrison	
2AShallNotBeInfringed	
GOPBetrayedAmerica	

Table B.5: Additional hashtags from the U.S. politics clusters in period 1 (February) and period 3 (April). Note that these hashtags are not ordered in any way by their relative usage.

Bibliography

[1] Mahdi Abavisani and Vishal M. Patel. Deep multimodal subspace clustering networks. *CoRR*, abs/1804.06498, 2018. URL http://arxiv.org/abs/1804.06498. 1.2.2, 1.2.2

[2] Margareta Ackerman and Shai Ben-David. Clusterability: A theoretical study. In David van Dyk and Max Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 1–8, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 4 2009. PMLR. URL http://proceedings.mlr.press/v5/ackerman09a.html. 2.2.4, 2.3

[3] Alberto Aleta and Yamir Moreno. Multilayer Networks in a Nutshell. *Annual Review of Condensed Matter Physics*, 10:45–62, 3 2019. doi: 10.1146/annurev-conmatphys-031218-013259. 1.2.2, 1.2.2, 1.2.2

[4] Esmaeil Alinezhad, Babak Teimourpour, Mohammad Mehdi Sepehri, and Mehrdad Kargari. Community detection in attributed networks considering both structural and attribute similarities: two mathematical programming approaches. *Neural Computing and Applications*, 2019. URL https://doi.org/10.1007/s00607-016-0526-5. 1.2.2

[5] Dmitri Alperovitch. Deep in thought: Chinese targeting of national security think tanks. Technical report, Crowdstrike, 2014. URL https://www.crowdstrike.com/blog/deep-thought-chinese-targeting-national-security-think-tanks/. 5.3

[6] Tahani Alqurashi and Wenjia Wang. Clustering ensemble method. *International Journal of Machine Learning and Cybernetics*, 10:1227 – 1246, 2018. URL https://doi.org/10.1007/s13042-018-0807-8. 1.2.2

[7] Shai Shalev-Shwartz anbd Shai Ben-David, editor. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2012. ISBN 978-1-107-05713-5. 1.2.1, 2.2.1, 2.2.4, 2.3

[8] Article19. Viral lies: Misinformation and the coronavirus. Technical report, Article19, 3 2020. URL https://www.article19.org/wp-content/uploads/2020/03/Coronavirus-briefing.pdf. 4.1

[9] S. Bai, Z. Zhou, J. Wang, X. Bai, L. J. Latecki, and Q. Tian. Automatic ensemble diffusion for 3d shape and image retrieval. *IEEE Transactions on Image Processing*, 28(1):88–101,

1 2019. ISSN 1057-7149. 1.2.1, 1.2.2, 3.1

[10] Song Bai, Shaoyan Sun, Xiang Bai, Zhaoxiang Zhang, and Qi Tian. Improving context-sensitive similarity via smooth neighborhood for object retrieval. *Pattern Recognition*, 83:353 – 364, 2018. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2018.06.001. URL http://www.sciencedirect.com/science/article/pii/S0031320318302115. 1.2.1, 1.2.2, 3.1

[11] Tadas Baltrusaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *CoRR*, abs/1705.09406, 2017. URL http://arxiv.org/abs/1705.09406. 1.2.2

[12] T. Baltrušaitis, C. Ahuja, and L. Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443, 2 2019. ISSN 0162-8828. doi: 10.1109/TPAMI.2018.2798607. 1.2.1, 1.2.2, 1.2.2, 1.2.2, 1.2.2

[13] Albert Laszlo Barabsi. *Network Science*. Cambridge University Press, 2015. URL http://networksciencebook.com/. 2.2.5, 3.1

[14] Shai Ben-David, Ulrike von Luxburg, and Dávid Pál. A sober look at clustering stability. In *COLT*, 2006. 1.2.1, 2.2.4

[15] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):10008, 10 2008. doi: 10.1088/1742-5468/2008/10/P10008. 1.2.2, 2.1.1, 5.2.1

[16] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, page 92–100, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 1581130570. doi: 10.1145/279943.279962. URL https://doi.org/10.1145/279943.279962. 2.1.1

[17] Svenja Boberg, Thorsten Quandt, Tim Schatto-Eckrodt, and Lena Frischlich. Pandemic Populism: Facebook Pages of Alternative News Media and the Corona Crisis – A Computational Content Analysis. *arXiv e-prints*, art. arXiv:2004.02566, 4 2020. 4.1

[18] Tossapon Boongoen and Natthakan Iam-On. Cluster ensembles: A survey of approaches with recent extensions and applications. *Computer Science Review*, 28:1 – 25, 2018. ISSN 1574-0137. doi: https://doi.org/10.1016/j.cosrev.2018.01.003. URL http://www.sciencedirect.com/science/article/pii/S1574013717300692. 1.2.2, 2.1, 2.2.2, 5.1.1, 5.2.2

[19] J. Roger Bray and J. T. Curtis. An ordination of the upland forest communities of southern wisconsin. *Ecological Monographs*, 27(4):325–349, 1957. doi: 10.2307/1942268. URL https://esajournals.onlinelibrary.wiley.com/doi/abs/10.2307/1942268. 5.3.1

[20] Piotr Bródka. A Method for Group Extraction and Analysis in Multilayer Social Networks. *arXiv e-prints*, art. arXiv:1612.02377, 12 2016. 1.2.2

[21] Ivan Brugere and Tanya Y. Berger-Wolf. Network model selection using task-focused minimum description length. *CoRR*, abs/1710.05207, 2017. URL http://arxiv.org/abs/1710.05207. 1.2.2

[22] Ivan Brugere, Chris Kanich, and Tanya Y. Berger-Wolf. Network model selection for task-focused attributed network inference. *CoRR*, abs/1708.06303, 2017. URL http://arxiv.org/abs/1708.06303. 1.2.2

[23] Ivan Brugere, Brian Gallagher, and Tanya Y. Berger-Wolf. Network structure inference, a survey: Motivations, methods, and applications. *ACM Comput. Surv.*, 51(2):24:1–24:39, 4 2018. ISSN 0360-0300. doi: 10.1145/3154524. URL http://doi.acm.org/10.1145/3154524. 2.1, 2.1, 2.1.1, 3.3

[24] Joaquín Calatayud, Rubén Bernardo-Madrid, Magnus Neuman, Alexis Rojas, and Martin Rosvall. Exploring the solution landscape enables more reliable network community detection. *Phys. Rev. E*, 100:052308, 11 2019. doi: 10.1103/PhysRevE.100.052308. URL https://link.aps.org/doi/10.1103/PhysRevE.100.052308. 6.3

[25] Gian Maria Campedelli, Iain Cruickshank, and Kathleen M. Carley. Detecting Latent Terrorist Communities Testing a Gower's Similarity-Based Clustering Algorithm for Multipartite Networks. In Luca Maria Aiello, Chantal Cherifi, Hocine Cherifi, Renaud Lambiotte, Pietro Lió, and Luis M. Rocha, editors, *Complex Networks and Their Applications VII*, volume 812, pages 292–303. Springer International Publishing, Cham, 2019. doi: 10.1007/978-3-030-05411-3_24. URL http://link.springer.com/10.1007/978-3-030-05411-3_24. 5.2.1, 6.3

[26] Gian Maria Campedelli, Iain Cruickshank, and Kathleen M. Carley. A complex networks approach to find latent clusters of terrorist groups. *Applied Network Science*, 4(1):59, 8 2019. ISSN 2364-8228. doi: 10.1007/s41109-019-0184-6. URL https://doi.org/10.1007/s41109-019-0184-6. 5.2.1, 5.2.1, 6.3

[27] CJ Carey. *GRAPH CONSTRUCTION FOR MANIFOLD DISCOVERY*. PhD thesis, University of Massachusetts Amherst, 2017. URL https://people.cs.umass.edu/~ccarey/pubs/thesis.pdf. 2.2.1

[28] Fabio Celli, F Marta L Di Lascio, Matteo Magnani, Barbara Pacelli, and Luca Rossi. Social Network Data and Practices: the case of Friendfeed. In *International Conference on Social Computing, Behavioral Modeling and Prediction*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010. 2.1

[29] Ping Chen, Lieven Desmet, and Christophe Huygens. A study on advanced persistent threats. In Bart De Decker and André Zúquete, editors, *Communications and Multimedia Security*, pages 63–72, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-44885-4. 5.1, 5.3

[30] Hong Cheng, Yang Zhou, and Jeffrey Xu Yu. Clustering large attributed graphs: A balance between structural and attribute similarities. *ACM Trans. Knowl. Discov. Data*, 5 (2), February 2011. ISSN 1556-4681. doi: 10.1145/1921632.1921638. URL https://doi.org/10.1145/1921632.1921638. 1.2.2, 2.1.1

[31] Catherine Cho, Sooyoung Kim, Jaewook Lee, and Dae-Won Lee. A tandem clus-

tering process for multimodal datasets. *European Journal of Operational Research*, 168(3):998 – 1008, 2006. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2004.05.020. URL http://www.sciencedirect.com/science/article/pii/S0377221704003935. Balancing Assembly and Transfer lines. 1.2.1, 1.2.2

[32] Petr Chunaev. Community detection in node-attributed social networks: a survey. *arXiv e-prints*, art. arXiv:1912.09816, 12 2019. 1.2.1, 1.2.2, 1.2.2, 1.2.2, 1.2.2, 2.1.1, 2.1.1, 2.2.2, 3.1

[33] Matteo Cinelli, Walter Quattrociocchi, Alessand ro Galeazzi, Carlo Michele Valensise, Emanuele Brugnoli, Ana Lucia Schmidt, Paola Zola, Fabiana Zollo, and Antonio Scala. The COVID-19 Social Media Infodemic. *arXiv e-prints*, art. arXiv:2003.05004, 3 2020. 4.1

[34] David Combe, Christine Largeron, Mathias Géry, and Előd Egyed-Zsigmond. I-louvain: An attributed graph clustering method. In Elisa Fromont, Tijl De Bie, and Matthijs van Leeuwen, editors, *Advances in Intelligent Data Analysis XIV*, pages 181–192, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24465-5. 1.2.2

[35] British Broadcasting Corporation. Li wenliang: Coronavirus kills chinese whistleblower doctor. *BBC News*, 2 2020. URL https://www.bbc.com/news/world-asia-china-51403795. 4.3.7

[36] Iain J. Cruickshank, Anthony Johnson, Timothy Davison, Matthew Elder, and Kathleen M. Carley. Detecting malware communities using socio-cultural cognitive mapping. In *2019 International Conference on Social Computing, Behavioral-Cultural Modeling, & Prediction and Behavior Representation in Modeling and Simulation*, 7 2019. 5.1.1, 5.2.1, 5.3.2, 5.3.4

[37] Manlio De Domenico, Andrea Lancichinetti, Alex Arenas, and Martin Rosvall. Identifying Modular Flows on Multilayer Networks Reveals Highly Overlapping Organization in Interconnected Systems. *Physical Review X*, 5(1):011027, 1 2015. doi: 10.1103/PhysRevX.5.011027. 1.2.2

[38] R. de Santiago and Luís C. Lamb. A ground truth contest between modularity maximization and modularity density maximization. *Artificial Intelligence Review*, 1 2020. URL https://link.springer.com/article/10.1007/s10462-019-09802-8. 3.1, 3.1, 3.3.2

[39] Daryl R. DeFord and Scott D. Pauls. Spectral clustering methods for multiplex networks. *Physica A: Statistical Mechanics and its Applications*, 533:121949, 2019. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2019.121949. URL http://www.sciencedirect.com/science/article/pii/S0378437119311379. 1.2.2

[40] Dell. Sakula malware family threat analysis, 7 2015. URL https://www.secureworks.com/research/sakula-malware-family. 5.3

[41] Mark E. Dickison, Matteo Magnani, and Luca Rossi. *Multilayer Social Networks*. Cambridge University Press, 2016. 1.2.2, 2.1.1

[42] Manlio De Domenico, V. Nicosia, Alexandre Arenas, and Vito Latora. Structural re-

ducibility of multilayer networks. *Nature communications*, 6:6864, 2015. 1.2.2

[43] M. Donoser and H. Bischof. Diffusion processes for retrieval revisited. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1320–1327, 6 2013. doi: 10.1109/CVPR.2013.174. 1.2.2, 2.1.2, 2.2.2, 5.2.2, 5.2.2

[44] Nicolas Dugué and Anthony Perez. Directed Louvain : maximizing modularity in directed networks. Research report, Université d'Orléans, 11 2015. URL `https://hal.archives-ouvertes.fr/hal-01231784`. 3.1

[45] F-Secure Labs. The dukes: 7 years of russian cyberespionage. Technical report, F-Secure Labs, 09 2015. 5.3, 5.3.4, 5.3.4

[46] Fangxiang Feng, Xiaojie Wang, and Ruifan Li. Cross-modal retrieval with correspondence autoencoder. In *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pages 7–16, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3063-3. doi: 10.1145/2647868.2654902. URL `http://doi.acm.org/10.1145/2647868.2654902`. 1.2.2

[47] Xiaoli Zhang Fern and Carla E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 36–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015414. URL `http://doi.acm.org/10.1145/1015330.1015414`. 1.2.2, 2.1.1, 2.2.2, 5.2.2, 5.2.2

[48] Emilio Ferrara. #COVID-19 on Twitter: Bots, Conspiracies, and Social Media Activism. *arXiv e-prints*, art. arXiv:2004.09531, 4 2020. 4.1

[49] Andreas Flache, Michael Mäs, Thomas Feliciani, Edmund Chattoe-Brown, Guillaume Deffuant, Sylvie Huet, and Jan Lorenz. Models of social influence: Towards the next frontiers. *Journal of Artificial Societies and Social Simulation*, 20(4):2, 2017. ISSN 1460-7425. doi: 10.18564/jasss.3521. URL `http://jasss.soc.surrey.ac.uk/20/4/2.html`. 5.2.2

[50] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Science*, 104(1):36–41, 1 2007. doi: 10.1073/pnas.0605965104. 3.1, 3.4.1

[51] Noah E. Friedkin and Eugene C. Johnsen. *Social Influence Network Theory: A Sociological Examination of Small Group Dynamics*. Structural Analysis in the Social Sciences. Cambridge University Press, 2011. doi: 10.1017/CBO9780511976735. 2.1.2, 2.1.2, 5.2.2, 5.3.6

[52] Riccardo Gallotti, Francesco Valle, Nicola Castaldo, Pierluigi Sacco, and Manlio De Domenico. Assessing the risks of "infodemics" in response to COVID-19 epidemics. *arXiv e-prints*, art. arXiv:2004.03997, 4 2020. 4.1

[53] Lucy L. Gao, Daniela Witten, and Jacob Bien. Testing for Association in Multi-View Network Data. *arXiv e-prints*, art. arXiv:1909.11640, 9 2019. 1.2.2

[54] Tennakoon Mudiyanselage Gayani Tennakoon, Khanh Luong, Wathsala Mohotti, Sharma Chakravarthy, and Richi Nayak. Multi-type relational data clustering for community de-

tection by exploiting content and structure information in social networks. In Abhaya C. Nayak and Alok Sharma, editors, *PRICAI 2019: Trends in Artificial Intelligence*, pages 541–554, Cham, 2019. Springer International Publishing. ISBN 978-3-030-29911-8. 1.2.2

[55] V. Gligorijević, Y. Panagakis, and S. Zafeiriou. Non-negative matrix factorizations for multiplex network analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):928–940, 4 2019. doi: 10.1109/TPAMI.2018.2821146. 1.2.2, 2.2.2

[56] Vladimir Gligorijević, Meet Barot, and Richard Bonneau. deepNF: deep network fusion for protein function prediction. *Bioinformatics*, 34(22):3873–3881, 06 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/bty440. URL `https://doi.org/10.1093/bioinformatics/bty440`. 1.2.2

[57] Derek Greene and Pádraig Cunningham. Producing a Unified Graph Representation from Multiple Social Network Views. *arXiv e-prints*, art. arXiv:1301.5809, January 2013. 2.1

[58] D. Guo, J. Zhang, X. Liu, Y. Cui, and C. Zhao. Multiple kernel learning based multi-view spectral clustering. In *2014 22nd International Conference on Pattern Recognition*, pages 3774–3779, 8 2014. doi: 10.1109/ICPR.2014.648. 1.2.1, 1.2.2

[59] R. Harakawa, S. Takimura, T. Ogawa, M. Haseyama, and M. Iwahashi. Consensus clustering of tweet networks via semantic and sentiment similarity estimation. *IEEE Access*, 7:116207–116217, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2936404. 1.2.2

[60] C. He, X. Fei, H. Li, Y. Tang, H. Liu, and Q. Chen. A multi-view clustering method for community discovery integrating links and tags. In *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*, pages 23–30, 11 2017. doi: 10.1109/ICEBE.2017.14. 1.2.2

[61] T. He, Y. Liu, T. H. Ko, K. C. C. Chan, and Y. Ong. Contextual correlation preserving multiview featured graph clustering. *IEEE Transactions on Cybernetics*, pages 1–14, 2019. ISSN 2168-2275. doi: 10.1109/TCYB.2019.2926431. 1.2.2

[62] Xin Hu and Kang G. Shin. Duet: Integration of dynamic and static analyses for malware clustering with cluster ensembles. In *Proceedings of the 29th Annual Computer Security Applications Conference*, ACSAC '13, pages 79–88, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2015-3. doi: 10.1145/2523649.2523677. URL `http://doi.acm.org/10.1145/2523649.2523677`. 5.1.1, 5.4

[63] Binxuan Huang. *Learning User Latent Attributes on Social Media*. PhD thesis, Carnegie Mellon University, 5 2020. 4.1, 4.2.1

[64] D. Huang, C. Wang, and J. Lai. Locally weighted ensemble clustering. *IEEE Transactions on Cybernetics*, 48(5):1460–1473, 5 2018. doi: 10.1109/TCYB.2017.2702343. 1.2.2, 2.2.2, 5.2.2, 5.2.2

[65] Dong Huang, Jian-Huang Lai, and Chang-Dong Wang. Combining multiple clusterings via crowd agreement estimation and multi-granularity link analysis. *Neurocomputing*, 170:240 – 250, 2015. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2014.05.094. URL `http://www.sciencedirect.com/science/article/pii/S0925231215005640`. Advances on Biological Rhythmic Pattern Generation: Exper-

iments, Algorithms and Applications Selected Papers from the 2013 International Conference on Intelligence Science and Big Data Engineering (IScIDE 2013) Computational Energy Management in Smart Grids. 1.2.2, 2.2.2

[66] Xiao Huang, Jundong Li, and Xia Hu. Label informed attributed network embedding. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, page 731–739, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450346757. doi: 10.1145/3018661.3018667. URL https://doi.org/10.1145/3018661.3018667. 2.1

[67] Y. Huang and H. Wangg. Consensus and multiplex approach for community detection in attributed networks. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 425–429, 12 2016. doi: 10.1109/GlobalSIP.2016.7905877. 3.3

[68] Yuming Huang, Ashkan Panahi, Hamid Krim, and Liyi Dai. Community Detection and Improved Detectability in Multiplex Networks. *arXiv e-prints*, art. arXiv:1909.10477, 9 2019. 1.2.2

[69] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 12 1985. doi: 10.1007/BF01908075. URL https://doi.org/10.1007/BF01908075. 2.2.1, 3.4, 4.3.4, 5.3.3

[70] Syed Fawad Hussain and Shariq Bashir. Co-clustering of multi-view datasets. *Knowledge and Information Systems*, 47:545–570, 2016. doi: 10.1007/s10115-015-0861-4. URL https://doi.org/10.1007/s10115-015-0861-4. 1.2.2, 2, 2.2.2

[71] Syed Fawad Hussain, Muhammad Mushtaq, and Zahid Halim. Multi-view document clustering via ensemble method. *J. Intell. Inf. Syst.*, 43(1):81–99, 8 2014. ISSN 0925-9902. doi: 10.1007/s10844-014-0307-6. URL https://doi.org/10.1007/s10844-014-0307-6. 1.2.2

[72] Wajahat Hussain. Role of social media in covid-19 pandemic. *The International Journal of Frontier Sciences*, 4, 4 2020. doi: 10.37978/tijfs.v4i2.144. URL http://publie.frontiersciencesassociates.com/index.php/tijfs/article/view/144. 4.1

[73] Insight. 3 sources dataset, 2009. URL http://mlg.ucd.ie/datasets/3sources.html. 2.1

[74] Lucas G. S. Jeub, Olaf Sporns, and Santo Fortunato. Multiresolution consensus clustering in networks. *Scientific Reports*, 8(3259), 2018. doi: 10.1038/s41598-018-21352-7. URL https://doi.org/10.1038/s41598-018-21352-7. 6.3

[75] Zhao Kang, Zipeng Guo, Shudong Huang, Siying Wang, Wenyu Chen, Yuanzhang Su, and Zenglin Xu. Multiple Partitions Aligned Clustering. *arXiv e-prints*, art. arXiv:1909.06008, 9 2019. 1.2.2

[76] Jon Kleinberg. An impossibility theorem for clustering. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, page 463–470, Cambridge, MA, USA, 2002. MIT Press. 2.2.1, 2.3

[77] Srijan Kumar, William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Community interaction and conflict on the web. *CoRR*, abs/1803.03697, 2018. URL http://arxiv.org/abs/1803.03697. 1.2.2

[78] Su Mon Kywe, Tuan-Anh Hoang, Ee-Peng Lim, and Feida Zhu. On recommending hashtags in twitter networks. In *Proceedings of the 4th International Conference on Social Informatics*, SocInfo'12, page 337–350, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 9783642353857. doi: 10.1007/978-3-642-35386-4_25. URL https://doi.org/10.1007/978-3-642-35386-4_25. 4.1, 4.2.1

[79] Zhen-zhong Lan, Lei Bao, Shoou-I Yu, Wei Liu, and Alexander G. Hauptmann. Multimedia classification and event detection using double fusion. *Multimedia Tools and Applications*, 71(1):333–347, 7 2014. ISSN 1573-7721. doi: 10.1007/s11042-013-1391-2. URL https://doi.org/10.1007/s11042-013-1391-2. 1.2.2

[80] Andrea Lancichinetti and Santo Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84:066122, 12 2011. doi: 10.1103/PhysRevE.84.066122. 2.1.1, 2.2.2, 3.1

[81] Andrea Lancichinetti and Santo Fortunato. Consensus clustering in complex networks. *Scientific Reports*, 2, 2012. URL https://doi.org/10.1038/srep00336. 1.2.2, 2.1.1

[82] Youwei Liang, Dong Huang, and Chang-Dong Wang. Consistency meets inconsistency: A unified graph learning framework for multi-view clustering. In *2019 IEEE International Conference on Data Mining (ICDM)*, 11 2019. 1.2.2, 2.2.2

[83] Chang Liu. Community detection in node attributed networks: A late-fusion approach. Master's thesis, University of Alberta, Alberta, 2019. 1.2.2

[84] Hongfu Liu, Zhiqiang Tao, and Yun Fu. Partition level constrained clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(10):2469–2483, 10 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2763945. URL https://doi.org/10.1109/TPAMI.2017.2763945. 2.1.1, 6.3

[85] Hongfu Liu, Zhiqiang Tao, and Zhengming Ding. Consensus Clustering: An Embedding Perspective, Extension and Beyond. *arXiv e-prints*, art. arXiv:1906.00120, 5 2019. 1.2.2

[86] Juncheng Lv, Zhao Kang, Boyu Wang, Luping Ji, and Zenglin Xu. Multi-view Subspace Clustering via Partition Fusion. *arXiv e-prints*, art. arXiv:1912.01201, 12 2019. 1.2.2

[87] Thomas Magelinski and Kathleen M. Carley. Community-based time segmentation from network snapshots: Streaming and holistic approaches for semi-static and dynamic node-sets. *Applied Network Science*, 4(25), 2019. doi: 10.1007/s41109-019-0136-1. URL https://doi.org/10.1007/s41109-019-0136-1. 4.3.4

[88] Markus Maier, Matthias Hein, and Ulrike von Luxburg. Optimal construction of k-nearest neighbor graphs for identifying noisy clusters. *arXiv e-prints*, art. arXiv:0912.3408, 12 2009. 2.2.1, 4.2.2

[89] Markus Maier, Ulrike von Luxburg, and Matthias Hein. How the result of graph clustering methods depends on the construction of the graph. *arXiv e-prints*, art. arXiv:1102.2075, 2

2011. 2.2.1, 2.2.5, 4.2.2

[90] Domenico Mandaglio, Alessia Amelio, and Andrea Tagarelli. Consensus community detection in multilayer networks using parameter-free graph pruning. *CoRR*, abs/1804.06653, 2018. URL http://arxiv.org/abs/1804.06653. 1.2.2, 1.2.2, 2, 2.2.2

[91] Daniel Marbach, James C. Costello, Robert Küffner, Nicci Vega, Robert J. Prill, Diogo M Camacho, Kyle R. Allison, Manolis Kellis, James J. Collins, and Gustavo Stolovitzky. Wisdom of crowds for robust gene network inference. In *Nature Methods*, 2012. 1.2.2

[92] Naoki Masuda and Petter Holme. Detecting sequences of system states in temporal networks. *Scientific Reports*, 9(795), 2019. doi: 0.1038/s41598-018-37534-2. URL https://doi.org/10.1038/s41598-018-37534-2. 4.3.4

[93] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878, 2010. ISSN 0036-8075. doi: 10.1126/science.1184819. URL https://science.sciencemag.org/content/328/5980/876. 1.2.2, 2.1.1, 2.1.1, 3.1, 6.3

[94] M. E. J. Newman. *Networks: An Introduction*. Oxford University Press, 2010. 1.2.1, 1.2.1, 2.1.1, 2.2.1, 2.2.5, 3.1, 3.1, 4.3.2, 5.2.1, 5.3.2, 6.3

[95] M. E. J. Newman. Community detection in networks: Modularity optimization and maximum likelihood are equivalent. *arXiv e-prints*, art. arXiv:1606.02319, 6 2016. 3.3, 3.3.1, 3.3.1, 3.4.1, 3.4.1

[96] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 849–856, Cambridge, MA, USA, 2001. MIT Press. URL http://dl.acm.org/citation.cfm?id=2980539.2980649. 2.1.1

[97] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. Multimodal deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 689–696, USA, 2011. Omnipress. ISBN 978-1-4503-0619-5. URL http://dl.acm.org/citation.cfm?id=3104482.3104569. 1.2.2

[98] Jingchao Ni, Hanghang Tong, Wei Fan, and Xiang Zhang. Flexible and robust multi-network clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, page 835–844, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336642. doi: 10.1145/2783258.2783262. URL https://doi.org/10.1145/2783258.2783262. 1.2.2

[99] Shraddha Pai and Gary D. Bader. Patient similarity networks for precision medicine. *Journal of Molecular Biology*, 430(18, Part A):2924 – 2938, 2018. ISSN 0022-2836. doi: https://doi.org/10.1016/j.jmb.2018.05.037. URL http://www.sciencedirect.com/science/article/pii/S0022283618305321. Theory and Application of

Network Biology Toward Precision Medicine. 1.2.2, 1.2.2, 1.2.2

[100] Shraddha Pai, Shirley Hui, Ruth Isserlin, Muhammad A Shah, Hussam Kaka, and Gary D Bader. netdx: interpretable patient classification using integrated patient similarity networks. *Molecular Systems Biology*, 15(3), 2019. doi: 10.15252/msb.20188497. URL http://msb.embopress.org/content/15/3/e8497. 1.2.2

[101] A. Roxana Pamfil, Sam D. Howison, Renaud Lambiotte, and Mason A. Porter. Relating modularity maximization and stochastic block models in multilayer networks. *CoRR*, abs/1804.01964, 2018. URL http://arxiv.org/abs/1804.01964. 1.2.2, 2.1.1, 3.3, 3.3.1, 3.3.1, 3.3.1, 3.4.1, 6.3

[102] Andreas Papadopoulos, Dimitrios Rafailidis, George Pallis, and Marios D. Dikaiakos. Clustering attributed multi-graphs with information ranking. In Qiming Chen, Abdelkader Hameurlain, Farouk Toumani, Roland Wagner, and Hendrik Decker, editors, *Database and Expert Systems Applications*, pages 432–446, Cham, 2015. Springer International Publishing. ISBN 978-3-319-22849-5. 1.2.2

[103] Andreas Papadopoulos, George Pallis, and Marios D. Dikaiakos. Weighted clustering of attributed multi-graphs. *Computing*, 99:813 – 840, 2017. URL https://doi.org/10.1007/s00607-016-0526-5. 1.2.2

[104] Lawrence Phillips, Chase Dowling, Kyle Shaffer, Nathan Oken Hodas, and Svitlana Volkova. Using social media to predict the future: A systematic literature review. *CoRR*, abs/1706.06134, 2017. URL http://arxiv.org/abs/1706.06134. 1.2.2

[105] V. Premachandran and R. Kakarala. Consensus of k-nns for robust neighborhood selection on graph-based manifolds. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1594–1601, 6 2013. doi: 10.1109/CVPR.2013.209. 2.2.1

[106] Lishan Qiao, Limei Zhang, Songcan Chen, and Dinggang Shen. Data-driven graph construction and graph learning: A review. *Neurocomputing*, 312:336 – 351, 2018. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2018.05.084. URL http://www.sciencedirect.com/science/article/pii/S0925231218306696. 1.2.1, 2.1, 3.3, 4.2.2

[107] Nimrod Rappoport and Ron Shamir. Multi-omic and multi-view clustering algorithms: review and cancer benchmark. *Nucleic Acids Research*, 46(20):10546–10562, 10 2018. ISSN 0305-1048. doi: 10.1093/nar/gky889. URL https://doi.org/10.1093/nar/gky889. 1.2.2, 1.2.2, 1.2.2, 3.3

[108] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Phys. Rev. E*, 74:016110, 7 2006. doi: 10.1103/PhysRevE.74.016110. URL https://link.aps.org/doi/10.1103/PhysRevE.74.016110. 3.1, 3.1

[109] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53 – 65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL http://www.sciencedirect.com/science/article/pii/0377042787901257. 2.2.1, 5.3.3

[110] J. Ruan. A fully automated method for discovering community structures in high dimensional data. In *2009 Ninth IEEE International Conference on Data Mining*, pages 968–973, 12 2009. doi: 10.1109/ICDM.2009.141. 2.2.1, 5.2.1, 6.3

[111] J. Saxe and K. Berlin. Deep neural network based malware detection using two dimensional binary program features. In *2015 10th International Conference on Malicious and Unwanted Software (MALWARE)*, pages 11–20, 2015. doi: 10.1109/MALWARE.2015.7413680. 5.1.1, 5.2, 5.1, 5.2.1, 5.2.1, 5.4

[112] Gregory D. Saxton, Jerome N. Niyirora, Chao Guo, and Richard D. Waters. #advocatingforchange: The strategic use of hashtags in social media advocacy. *Advances in Social Work*, 16(1), 7 2015. doi: 10.18060/17952. URL https://advancesinsocialwork.iupui.edu/index.php/advancesinsocialwork/article/view/17952. 4.1

[113] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93, 9 2008. doi: 10.1609/aimag.v29i3.2157. URL https://www.aaai.org/ojs/index.php/aimagazine/article/view/2157. 2.1

[114] Allison Shapp. Variation in the use of twitter hashtags. Master's thesis, New York University, 2014. URL https://www.nyu.edu/projects/shapp/Shapp_QP2_Hashtags_Final.pdf. 4.1

[115] Pavica Sheldon, Erna Herzfeldt, and Philipp A. Rauschnabel. Culture and social media: the relationship between cultural values and hashtagging styles. *Behaviour & Information Technology*, 0(0):1–13, 2019. doi: 10.1080/0144929X.2019.1611923. URL https://doi.org/10.1080/0144929X.2019.1611923. 4.1, 4.2.1

[116] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *CoRR*, abs/1708.01967, 2017. URL http://arxiv.org/abs/1708.01967. 1.2.2

[117] Kai Shu, Deepak Mahudeswaran, and Huan Liu. Fakenewstracker: a tool for fake news collection, detection, and visualization. *Computational and Mathematical Organization Theory*, 25(1):60–71, 3 2019. ISSN 1572-9346. doi: 10.1007/s10588-018-09280-3. URL https://doi.org/10.1007/s10588-018-09280-3. 1.2.2

[118] Nitish Srivastava and Ruslan Salakhutdinov. Learning representations for multimodal data with deep belief nets. In *ICML Representation Learning Workshop*, 2012. 1.2.2

[119] Alexander Strehl and Joydeep Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, March 2003. ISSN 1532-4435. doi: 10.1162/153244303321897735. URL https://doi.org/10.1162/153244303321897735. 1.2.2, 2.2.2, 5.1.1

[120] Symantec. Advanced persistent threats: A symantec perspective. Technical report, Symantec, 2012. URL https://www.symantec.com/content/en/us/enterprise/white_papers/b-advanced_persistent_threats_WP_21215957.en-us.pdf. 5, 5.1

[121] Andrea Tagarelli, Alessia Amelio, and Francesco Gullo. Ensemble-based community detection in multilayer networks. *Data Mining and Knowledge Discovery*, 31:1506 – 1543, 2017. URL https://doi.org/10.1007/s10618-017-0528-8. 1.2.2, 2, 2.1.1

[122] Aditya Tandon, Aiiad Albeshri, Vijey Thayananthan, Wadee Alhalabi, and Santo Fortunato. Fast consensus clustering in complex networks. *Physical Review E*, 99(4):042301, 4 2019. doi: 10.1103/PhysRevE.99.042301. 1.2.2, 2.1.1, 2.1.1, 2.2.2

[123] Lei Tang and Huan Liu. *Community Detection and Mining in Social Media*. Morgan & Claypool Publishers, 2010. 1.2.2, 1.2.2, 1.2.2, 2.1.1, 2.2.2

[124] Hong Tao, Chenping Hou, Xinwang Liu, Dongyun Yi, and Jubo Zhu1. Reliable multi-view clustering. In *Proceedings of The Thirty-Second AAAI Conference on Artificial Intelligence*, AAAI-18. AAAI Press, 2018. 1.2.2, 2.1.2

[125] Zhiqiang Tao, Hongfu Liu, Sheng Li, Zhengming Ding, and Yun Fu. From ensemble clustering to multi-view clustering. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 2843–2849. AAAI Press, 2017. ISBN 9780999241103. 1.2.2, 6.3

[126] Zhiqiang Tao, Hongfu Liu, Sheng Li, Zhengming Ding, and Yun Fu. Robust spectral ensemble clustering via rank minimization. *ACM Trans. Knowl. Discov. Data*, 13(1), January 2019. ISSN 1556-4681. doi: 10.1145/3278606. URL https://doi.org/10.1145/3278606. 1.2.2

[127] V. A. Traag, P. van Dooren, and Y. Nesterov. Narrow scope for resolution-limit-free community detection. *Physical Review E*, 84(1):016114, 7 2011. doi: 10.1103/PhysRevE.84.016114. 3.1, 3.1, 3.1, 3.3.2

[128] V. A. Traag, G. Krings, and P. van Dooren. Significant scales in community structure. *Scientific Reports*, 3, 7 2013. URL https://www.nature.com/articles/srep02930#citeas. 3.1, 3.4.1

[129] V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Nature: Scientific Reports*, 9, 2019. doi: https://doi.org/10.1038/s41598-019-41695-z. URL https://www.nature.com/articles/s41598-019-41695-z. 2.1.1, 2.2.1

[130] Ankita Verma and Kamal K. Bharadwaj. Identifying community structure in a multi-relational network employing non-negative tensor factorization and ga k-means clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(1), 2 2017. ISSN 1942-4795. doi: 10.1002/widm.1196. URL https://doi.org/10.1002/widm.1196. 2.2.2

[131] Carlos Vicient and Antonio Moreno. Unsupervised topic discovery in micro-blogging networks. *Expert Systems with Applications*, 42(17):6472 – 6485, 2015. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2015.04.014. URL http://www.sciencedirect.com/science/article/pii/S0957417415002444. 4.1, 4.2.1, 4.2.2, 4.3.6

[132] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.*, 11:2837–2854, 12 2010. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1756006.1953024. 2.2.1, 3.4, 5.3.3

[133] Ulrike von Luxburg. A Tutorial on Spectral Clustering. *arXiv e-prints*, art. arXiv:0711.0189, November 2007. 2.1.1

[134] B. Wang, J. Jiang, W. Wang, Z. Zhou, and Z. Tu. Unsupervised metric fusion by cross diffusion. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2997–3004, 6 2012. doi: 10.1109/CVPR.2012.6248029. 1.2.2, 2.1.1, 2.1.2, 5.2.2, 5.2.2

[135] Bo Wang, Aziz M. Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature Methods*, 11:333–337, 2014. 1.2.2, 2.1.2

[136] Bo Wang, Armin Pourshafeie, Marinka Zitnik, Junjie Zhu, Carlos D. Bustamante, Serafim Batzoglou, and Jure Leskovec. Network enhancement as a general method to denoise weighted biological networks. *Nature Communications*, 9:3108, 8 2018. doi: 10.1038/s41467-018-05469-x. 1.2.2, 5.3.6

[137] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994. 6.3

[138] William H. Weir, Scott Emmons, Ryan Gibson, Dane Taylor, and Peter J. Mucha. Post-processing partitions to identify domains of modularity optimization. *arXiv e-prints*, art. arXiv:1706.03675, 6 2017. 3.4.1, 6.3

[139] Jianlong Wu, Zhouchen Lin, and Hongbin Zha. Essential tensor learning for multi-view spectral clustering. *CoRR*, abs/1807.03602, 2018. URL http://arxiv.org/abs/1807.03602. 1.2.2, 2.2.2

[140] Jie Wu, Wenzhang Zhuge, Hong Tao, Chenping Hou, and Zhao Zhang. Incomplete multi-view clustering via structured graph learning. In Xin Geng and Byeong-Ho Kang, editors, *PRICAI 2018: Trends in Artificial Intelligence*, pages 98–112, Cham, 2018. Springer International Publishing. ISBN 978-3-319-97304-3. 1.2.2

[141] Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*, 2014. 1.2.1, 1.2.2

[142] Feng Xiao, Tomoya Noro, and Takehiro Tokuda. Finding news-topic oriented influential twitter users based on topic related hashtag community detection. *J. Web Eng.*, 13(5–6): 405–429, November 2014. ISSN 1540-9589. 4.1, 4.2.2

[143] Zhiqiang Xu and Yiping Ke. Effective and efficient spectral clustering on text and link data. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, page 357–366, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450340731. doi: 10.1145/2983323.2983708. URL https://doi.org/10.1145/2983323.2983708. 1.2.2, 2.2.4

[144] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. Network representation learning with rich text information. In *Proceedings of the 24th Interna-*

tional Conference on Artificial Intelligence, IJCAI'15, page 2111–2117. AAAI Press, 2015. ISBN 9781577357384. 2.1

[145] Kai-Cheng Yang, Christopher Torres-Lugo, and Filippo Menczer. Prevalence of Low-Credibility Information on Twitter During the COVID-19 Outbreak. *arXiv e-prints*, art. arXiv:2004.14484, 4 2020. 4.1

[146] X. Yang, L. Prasad, and L. J. Latecki. Affinity learning with diffusion on tensor product graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):28–38, 1 2013. ISSN 0162-8828. 2.1.2, 2.1.2, 5.3.6

[147] Y. Yang and H. Wang. Multi-view clustering: A survey. *Big Data Mining and Analytics*, 1(2):83–107, 6 2018. ISSN 2096-0654. 1.2.2, 1.2.2, 1.2.2, 1.2.2, 2.1, 3.3

[148] Fanghua Ye, Zitai Chen, Hui Qian, Rui Li, Chuan Chen, and Zibin Zheng. New approaches in multi-view clustering. *Recent Applications in Data Clustering*, 2018. doi: 10.5772/intechopen.75598. URL https://www.intechopen.com/books/recent-applications-in-data-clustering/new-approaches-in-multi-view-clustering. 1.2.2, 3.3

[149] Yanfang Ye, Tao Li, Donald Adjeroh, and S. Sitharama Iyengar. A survey on malware detection using data mining techniques. *ACM Comput. Surv.*, 50(3):41:1–41:40, 6 2017. ISSN 0360-0300. doi: 10.1145/3073559. URL http://doi.acm.org/10.1145/3073559. 5.1, 5.1.1

[150] H. Yu, Y. Lian, L. Zong, and L. Tian. Self-paced learning based multi-view spectral clustering. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 6–10, 11 2017. doi: 10.1109/ICTAI.2017.00013. 1.2.2, 2.2.2, 2.2.4, 3.1, 4.2.2, 6.3

[151] C. Zhang, H. Fu, Q. Hu, X. Cao, Y. Xie, D. Tao, and D. Xu. Generalized latent multi-view subspace clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42 (1):86–99, 1 2020. ISSN 1939-3539. doi: 10.1109/TPAMI.2018.2877660. 1.2.2

[152] Mimi Zhang. Weighted Clustering Ensemble: A Review. *arXiv e-prints*, art. arXiv:1910.02433, 10 2019. 1.2.2, 2.1

[153] Y. Zhang, W. Yang, B. Liu, G. Ke, Y. Pan, and J. Yin. Multi-view spectral clustering via tensor-svd decomposition. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 493–497, 11 2017. doi: 10.1109/ICTAI.2017.00081. 1.2.2

[154] Yang Zhang. Language in our time: An empirical analysis of hashtags. In *The World Wide Web Conference*, WWW '19, page 2378–2389, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313480. URL https://doi.org/10.1145/3308558.3313480. 4.1, 4.2.1

[155] Cangqi Zhou, Liang Feng, and Qianchuan Zhao. A novel community detection method in bipartite networks. *Physica A: Statistical Mechanics and its Applications*, 492: 1679 – 1693, 2018. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2017.11.089. URL http://www.sciencedirect.com/science/article/pii/

S0378437117311470. 2.1.1, 5.2.2, 5.3.4

[156] Jie Zhou, Hongchan Zheng, and Lulu Pan. Ensemble clustering based on dense representation. *Neurocomputing*, 357:66 – 76, 2019. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2019.04.078. URL http://www.sciencedirect.com/science/article/pii/S0925231219306794. 1.2.2, 2.2.2

[157] Yu Zhou, Xiang Bai, Wenyu Liu, and Longin Jan Latecki. Similarity fusion for visual tracking. *International Journal of Computer Vision*, 118(3):337–363, 7 2016. ISSN 1573-1405. doi: 10.1007/s11263-015-0879-9. URL https://doi.org/10.1007/s11263-015-0879-9. 1.2.2

[158] X. Zhu, C. C. Loy, and S. Gong. Constructing robust affinity graphs for spectral clustering. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1450–1457, 6 2014. doi: 10.1109/CVPR.2014.188. 4.2.2

[159] W. Zhuang, Y. Ye, Y. Chen, and T. Li. Ensemble clustering for internet security applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1784–1796, 2012. 5.1.1

[160] Marinka Zitnik, Francis Nguyen, Bo Wang, Jure Leskovec, Anna Goldenberg, and Michael M. Hoffman. Machine Learning for Integrating Data in Biology and Medicine: Principles, Practice, and Opportunities. *arXiv e-prints*, art. arXiv:1807.00123, 6 2018. 1.2.2, 1.2.2, 1.2.2, 1.2.2, 1.2.2