Multi-view based Approaches for Collective Learning and Retrieval Tasks

A THESIS

submitted by

S. SHIVASHANKAR

for the award of the degree

of

MASTER OF SCIENCE
(by Research)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS.

September 2011

THESIS CERTIFICATE

This is to certify that the thesis entitled **Multi-view based Approaches for Collective Learning and Retrieval Tasks**, submitted by **S. Shivashankar**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Science (by Research)**, is a bona fide record of the research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. B. Ravindran
Research Guide
Associate Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place: Chennai

Date:

ACKNOWLEDGEMENTS

I would like to thank everyone with whom I have interacted and learned useful things of different magnitude. In this section, I will attempt to mention everyone (atleast most of them) who have played a constructive role in my Masters work. First and foremost, Dr. B Ravindran, for his support and guidance. Dr. Ashish V Tendulkar, again for his support and guidance. I would like to specially thank Dr. BR and Dr. AVT for their help and care in all the issues. I am grateful to the course instructors who helped me improving my learning curve on various aspects of the subject. GTC members - Dr. Sutanu Chakraborty, Dr. Ramanathan and Dr. Kamala Kirthivasan, for their valuable comments on my research work. Collaborators - Dr. Anil Maddulapalli, Dr. AVT, Dr. BR, Dr. Srinivasa Raghavan and Srivathsan, for sharing the workload and being a part of the journey. To all friends and RISE Lab members, especially to Abhishek, Deepak, Karthick, Pradyot, Priya, Ranga, Sadagopan and Yousuf for their help and support. Last and never the least, I would like to thank General Motors India Science Lab, for funding and supporting my research work.

ABSTRACT

KEYWORDS: Collective Classification, Multi-view Learning, Multi-view IR, Sentiment Analysis, Linked Data Analysis, Protein Structure Similarity.

Machine learning techniques can be used to build systems to perform various tasks such as classification, clustering, retrieval, etc. In traditional machine learning algorithms, the data points are represented in a d-dimensional space which defines a *view* of the data. But many real-world datasets possess rich additional information that can be leveraged to improve the system's performance, reduce labeling cost, etc. For example, in webpage classification the content of the webpage and text on hyperlink pointing to this page can be used as two different views. Multi-view learning is a technique which allows the use of multiple views where a model built on one of the views can aid learning the models built on other views. It has been shown that multi-view approaches are more effective than single-view approaches for various domains. In this work, we propose multi-view based approaches for the following two tasks:

- *Linked Data Classification:* In general, all the machine learning paradigms assume the data points to be independent and identically distributed (i.i.d). But many real-world datasets such as webpages, images and protein interaction networks possess additional link information which provides local label smoothness. For example, in webpage classification, links between webpages convey that there is a strong correlation between labels of the linked pages. This can be exploited to provide regularization and thereby improve the performance. Collective classification is one of the popular approaches that can handle this type of network data. It combines traditional machine learning and link based classification in an iterative procedure. This involves two important steps: 1) collective learning – training the base classifier on

the appended content and link information with the given (partially) labeled network data; and 2) collective inference – applying the trained base classifier on the partially labeled network data and labeling it completely. Most of the collective classification frameworks assume that sufficient labeled data is available for training the base classifier. The trained base classifier is applied on the test data iteratively to perform collective inference. But, there has been little attention paid towards collective learning on partially labeled network data. We propose a multi-view learning based algorithm which utilizes both content and link views effectively to learn from the unlabeled data also. We empirically evaluate the effectiveness of the proposed framework by showing that it performs better than traditional single view approaches on sentiment analysis and standard linked datasets such as Cora, Citeseer and WebKB. We also performed sentiment analysis on automobile reviews.

- *Similar Protein Structures Retrieval:* With the rapid expansion of protein structure databases, the task of efficiently retrieving structures similar to a given protein has gained importance. It involves two major issues: (i) an effective protein structure representation that captures inherent relationship between fragments and facilitates efficient comparison between structures, (ii) an effective framework to accommodate *different* retrieval requirements. Recently, researchers proposed a vector space model of proteins using *bag of fragments* representation (FragBag), which corresponds to a basic information retrieval model. In this work, first, we propose an improved representation of protein structures using Latent Dirichlet Allocation (LDA) topic model. Second, we handle the most important requirement which is to retrieve proteins, whether they are close or intermediate or remote homologs. Close, remote and intermediate homologs are defined based on Structural Alignment Score (SAS). The SAS thresholds typically are 2, 3.5 and 5 for close, intermediate and remote homologs respectively. In order to meet diverse objectives, we propose a multi-view based framework that combines multiple representations and retrieval techniques. We experimentally show that the proposed multi-view based retrieval framework performs better than state-of-the-art filter and match techniques across different retrieval requirements. The experiments are performed on FragBag dataset, which contains 2,930 sequence non-redundant structures selected from CATH version 2.4.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AUC	Area Under the Curve
BoW	Bag of Words
CATH	Class, Architecture, Topology and Homology level classification
CPU	Central Processing Unit
EM	Expectation Maximization
FragBag	Bag of Fragments
i.i.d	Independent and identically distributed
IR	Information Retrieval
ISCL	Iterative Structured Collective Learning
KDD	Knowledge Discovery and Data mining
KL	Kullback-Leibler
KNN	K-Nearest Neighbor
LDA	Latent Dirichlet Allocation
MGSA	Multi-grain Sentiment Analysis
MVCL	Multi-view Collective Learning
NB	Naive Bayesian
NLP	Natural Language Processing
ROC	Receiver Operating Characteristic
SAS	Structural Alignment Score
SSL	Semi-Supervised Learning
TF	Term Frequency
TF-IDF	Term Frequency-Inverse Document Frequency

CHAPTER 1

INTRODUCTION

Machine learning techniques can be used to build systems that perform various tasks such as classification, clustering and retrieval in an automated fashion. The class of machine learning algorithms used varies depending on the task performed and the type of experience provided to the learning agent. In traditional machine learning algorithms, the data points are represented in a d-dimensional space which defines a *view* of the data. Much work has been done on finding a suitable representation for the data. A representation can be generated by choosing a subset of dimensions or by projecting the features onto a different space. The suitability of a representation is measured with respect to certain criteria on the features of the input data such as information gain, variance or with respect to the target task. These techniques are commonly referred to as pre-processing methods and they are part of the 'data transformation' stage of Knowledge Discovery and Data mining (KDD) process. There is another class of methods which instead of selecting a single most suitable view, focuses on simultaneously utilizing multiple views. For example, a protein can be represented using its structural and amino acid sequence information. It would be effective to use both the representations together rather than using either of them separately or using a single combined representation. In this work, we propose methods that utilize multiple views for linked data classification and similar protein structure retrieval tasks. In the literature, researchers refer to multiple views as *multi-view* or *multi-viewpoints*. We use *multi-view* throughout the text. In the following section, we give the taxonomy of common machine learning strategies.

1.1 Overview of Machine Learning Strategies

Taxonomy of common machine learning paradigms is given below :

- *Supervised learning* techniques learn a function to map the given input to discrete/continuous output space. For example, in a webpage classification problem, the learner approximates a function mapping the feature vector into "Politics" or "Cricket" by reducing the error on training input-output examples. Since the function outputs a discrete value, it is referred to as classification. Consider another example, where the input review text has to be mapped to a score in between 1 and 10. In this case, the output is a continuous value, it is referred to as regression.

  Active Learning is a supervised learning paradigm, which allows querying the user/oracle to obtain the label for an input datapoint. Typically, it is used in the cases where there is a fixed budget for getting labels. The querying must be done on the most informative datapoints with respect to the task in hand.

- *Unsupervised learning* techniques model the input data. The input does not include class labels as in the case of supervised learning. For example, to understand user browsing behaviour, we would like to group users based on their online browsing pattern and model each group. It is also called as clustering. The grouping is performed with respect to an objective function. For example, in k-means algorithm clustering is done such that the datapoints within a cluster are more similar and the datapoints across clusters are less similar. Other unsupervised learning techniques include association rule mining, blind separation techniques for feature extraction such as principal component analysis, independent component analysis etc.

- *Semi-supervised learning (SSL)* combines both labeled and unlabeled data to learn an appropriate function for prediction. Typically, it uses a small amount of labeled data and large amount of unlabeled data. It's a relatively new paradigm, and is gaining importance due to the fact that getting access to completely labeled data is hard and costly. Use of unlabeled data provides regularization, and it has been shown to improve the performance.

  Transductive learning works only on the labeled and unlabeled training data, and cannot handle unseen data. On the other hand, inductive learners can handle unseen data.

- *Reinforcement learning* is a machine learning paradigm, that falls neither under supervised nor unsupervised learning methods. An RL agent interacts with the environment and receives rewards for every action performed. An action can be chosen from the action space based on different criteria. Based on the experience gained over repeated runs, the agent computes the desirability

of choosing a particular action. The agent looks to maximize the reward accumulated over time, i.e., it looks to choose actions that promise higher returns in the long run. Unlike unsupervised learning, there is a notion of feedback in terms of rewards and next state.

Typically all the machine learning techniques assume the datapoints to have only one representation, which is also called as *single view*. But many real-world datasets possess additional information that can be utilized to improve the performance. For example, in text classification, sentences in a document can be represented using 'bag of words' representation which captures the lexical information. Alternatively, the sentences can also be represented using its 'parse tree' which captures the syntactic information. Rather than using either of the views (representations) separately or using a single representation by combining both views, it might be effective to use both views together where a model built on one of the views can aid learning the model built on other view. A relatively new machine learning technique to handle multiple views is called as *Multi-view Learning*[6; 31]. It has been successfully used in conjunction with traditional paradigms such as supervised, unsupervised and semi-supervised approaches[2; 6; 13; 15; 31; 35; 52; 55; 56; 65]. Researchers have shown that utilizing multiple views together is more efficient than using a single view alone or using a combined representation of multiple views.

1.2 Motivation And Objectives

Across domains, many machine learning problems involve data which naturally comprises multiple views. *Multi-view Learning* is a machine learning technique that can utilize multiple views in a supervised, semi-supervised or unsupervised setup. Seminal work in this field was done by Blum and Mitchell[6], where they proposed a "co-training" based multi-view learning algorithm which bootstraps a set of classifiers from high confidence labels. In that work, they assumed the views

to be self-sufficient [1] and conditionally independent given the class label. In other words, they assume a natural split of features into multiple views. For example, in web page classification, an input web page can be classified using its content *or* the text on hyperlinks pointing to this page. In object detection, an object can be classified using its color *or* shape. In a multi-modal setting, multiple views can be defined on signals from separate input sensors. For example, in biometrics, a person's identity can be found using his fingerprint *or* iris scan inputs. It is impractical to expect such natural splits in all the domains. So, researchers started working towards finding suitable artificial feature splits and necessary conditions for multi-view learning to be effective. Nigam and Ghani[31] presented other SSL algorithms for multi-view learning, which includes co-EM and self-training. They also analysed the conditions necessary for "co-training" to be successful. Following Nigam and Ghani, many approaches for multi-view learning[35; 52; 55; 56; 65] exploit multiple redundant views to effectively learn a set of classifiers defined using each of the views. Multi-view learning is found to be advantageous compared to single-view learning especially when the weaknesses of one view complement the strengths of the other[6; 13; 55]. This forms the basic motivation for our proposed multi-view based methods for linked data classification and similar protein structures retrieval tasks. In both the tasks, we exploit redundant views to improve the performance.

In brief, the approaches can be described as follows :

- In the first part of this work, we focus on multi-view methods for linked data classification. One of the popular frameworks that deals with linked data classification is *collective classification*. Collective classification approaches utilize both node's attributes (content) and link information in an iterative procedure[27; 48; 59]. Typically, the feature vector based on link information is the class distribution of neighboring nodes[27; 59]. Since acquiring fully labeled data is not possible in many domains, we focus on collective classification in a semi-supervised setup. In a partially labeled graph, many nodes need not be labeled. The feature vector for link view would end up

[1] A view is said to be self-sufficient, if classification can be performed using that view alone.

being incomplete. It can even turn out to be zero vectors. So we refer to the link view as weaker view. On the other hand, content view is self-sufficient. Hence collective classification on partially labeled graphs involves stronger content and weaker link views. We propose a multi-view based framework which utilizes the stronger view to bootstrap and learn model on the weaker view. And, utilizes the weaker view to help stronger view only when it is self-sufficient. Since link view is bootstrapped and modified by content view, link view can be seen as a redundant representation of the content view itself. We empirically evaluate the multi-view methods on sentiment analysis and other standard linked datasets such as Cora, Citeseer, WebKB[48].

- In the second part of this work, we propose a multi-view based framework for similar protein structures retrieval task. It is an important requirement to retrieve proteins that are similar to a given query protein, whether they are close or intermediate or remote homologs. Close, remote and intermediate homologs are defined based on Structural Alignment Score (SAS) that measures structural similarity between proteins. The SAS thresholds are 2, 3.5 and 5 for close, intermediate and remote homologs respectively [25]. An efficient protein structure comparison method should perform well across different retrieval requirements. In order to handle this, we exploit multiple redundant representations of the protein fragments. We show that the proposed multi-view based protein structure retrieval technique is more effective across different requirements than single view based methods. We evaluate the multi-view protein structures retrieval framework on FragBag dataset [25].

1.3 Contributions of the Thesis

Key contributions of this thesis are:

- We propose a multi-view framework to solve collective classification, where we treat content and link as two different views.
- To the best of our knowledge, no framework has been proposed for sentiment analysis in a semi-supervised environment which utilizes domain knowledge to build the graph.
- We propose a novel representation for protein structures, where we have adapted powerful models from statistical NLP literature to solve the task.
- We have also proposed a novel multi-view based retrieval framework which exploits multiple redundant representations of protein structure.

1.4 Organization

In this work, we propose multi-view based approaches for linked data classification and similar protein structures retrieval tasks. In Chapter 2, we provide a detailed literature review for the target tasks. In Chapter 3, we explain the multi-view framework for linked data classification. This chapter includes problem setup, proposed approach and experimental results on sentiment analysis and other standard linked datasets. In Chapter 4, we explain the multi-view framework for similar protein structures retrieval task. This chapter includes problem setup, proposed approach and experimental results on FragBag dataset [25]. In Chapter 5, we provide concluding remarks on the study and elaborate on the future work that could follow the current work.

CHAPTER 2

Background and Related work

In this chapter, we provide a detailed literature survey of the techniques proposed for collective classification and similar protein structures retrieval task.

2.1 Linked Data Classification

Classical Machine Learning techniques assume the data to be i.i.d, but the real world data is inherently relational and can generally be represented using graphs or some variants of them. The importance of modelling structured data is evident from its increasing presence : WWW, social networks, organizational networks, images, protein sequences, relational data, etc. This field has recently been receiving a lot of attention in the community under different themes depending on the problem addressed and the nature of solution proposed. Researchers in each of these different areas have proposed very useful and successful frameworks. Some of them include the following:

- *Collective classification* involves use of a local classifier that embeds the node's own attributes and neighbors' information in a feature vector, and classifies the nodes in an iterative procedure[27; 48; 59].
- *Statistical Relational Learning* combines statistics (uncertainty) and relational information (first-order logic) to model the target domain[4].
- *Structured prediction* involves the use of machine learning techniques on structured objects that embed the relationship between output classes[18; 72].
- *Graph based SSL techniques* apply the local smoothness property on the structured data, i.e., uses it as a constraint or side-information[16; 43; 70; 71].
- *Kernel methods for structured data* deal with developing similarity functions for objects of a domain that can handle relationships between objects, as well as heterogeneous representations[60].

There has been significant research progress on these subtasks in each area individually. Several workshops are held for each of these fields such as SRL, ILP, StarAI, GBR, GDM. In this thesis, we focus on *collective classification*. Collective classification algorithms deal with within-network classification by simultaneously labelling a set of related nodes, allowing estimates of neighboring labels to influence one another. It involves two sub-tasks : collective learning and collective inference. 1) Collective learning deals with training the base classifier on the appended content and link information with the given (partially) labeled network data; and 2) collective inference deals with applying the trained base classifier on the partially labeled network data and labeling it completely. There has been huge increase in interest on collective inference since it is a common factor to all these subtasks. Some of the workshops on collective inference are "CVPR WS on Inference in Graphical Models and Structured Potentials", "Propagation Algorithms on Graphs with Cycles: Theory and Applications", "Approximate inference - How far have we come?". Though collective inference is an important subissue, collective learning on network data involves a lot of (new) challenges posed by various domains. Some of them are mentioned below:

- How to learn the prediction model (classification/regression) with sparsely labeled network data?
- How to handle severe class imbalance in datasets?
- How to handle streaming data?
- How to handle large scale data?
- How to leverage additional (domain, side) information to reduce the complexity of estimation?

These new challenges have motivated researchers to work towards collective learning techniques that can overcome the existing limitations[10; 57]. Some of the workshops which focus on collective learning in addition to inference include "MLG : Mining and Learning with Graphs", "CoLISD : Collective Learning and

Inference on Structured Data" and "Approximate Learning of Large Scale Graphical Models: Theory and Applications". In this work, we address the first challenge i.e., collective learning on sparsely labeled networks.

Most of the collective classification frameworks assume that sufficient labeled data is available to train the base classifier[7; 21; 27; 48; 59]. They build a feature vector by appending the content and link information. The learnt model is used for inferencing on partially labeled network using an iterative framework such as Iterative Collective Algorithm (ICA), Gibbs Sampling, Belief Propagation, etc. One of the earliest seminal papers in this field was proposed by Chakrabarti et al. on using hyperlinks for hypertext classification[59]. Recent works include cautious collective classification procedures, where labels of the k most confidently predicted nodes are committed in each iteration[33]. This is referred to as a "cautious" procedure, since in collective classification labels of all the unlabeled nodes are updated in each iteration. One of the major advantages of collective classification is its ability to learn different types of dependencies in a network such as positive vs. negative correlation, different degrees of correlation, etc. This advantage marks its difference from graph based semi-supervised learning techniques which can also be used to solve within-network classification[16; 70; 71]. Graph based SSL techniques assume homophily, i.e, neighboring nodes should have same labels. The method would fail in cases where this assumption does not hold good. However, when the labeled data is very sparse, the performance of collective classification might degrade due to lack of sufficiently labeled neighbors[28]. There is very little attention paid towards collective learning frameworks for partially labeled network data[10; 49]. Acquiring sufficient amount of labeled data is not possible in many cases. Also, it is very costly to label them manually. Fortunately, many datasets are naturally partially labeled. For example, in the task of web page classification, it is possible to get partial labels on web pages using social bookmarking sites, open directory project, etc. The task is to label the unlabeled objects with one or more categories from a finite set of categories and learn a clas-

sification model that can be used for collective inference. Lu and Getoor proposed an *EM-Like* collective learning approach which builds classifiers on content and link information separately, and combines the results by multiplying the probability distribution given by two classifiers. Work by Gallagher et al.[10] deals with sparsely labeled graphs by adding "ghost edges" into the network. It is done by identifying key unlabeled nodes that would influence the overall performance and connecting it to labeled nodes by analysing network properties. They deal with networks where the nodes do not have attributes. But, in this work, we are interested in collective learning on partially labeled networks which have both content and link information, i.e, nodes have their own attributes. We handle bootstrapping and learning as two different steps, and not as a single step as in the case of traditional collective classification approaches including the *EM-Like* procedure[49]. Since bootstrapping procedure defines the link view along with the given labeled data, we propose a cautious procedure for bootstrapping the labels of unlabeled nodes.

2.2 Similar Protein Structures Retrieval

Several methods have been proposed in literature for protein structure comparison. These methods compare a pair of protein structures, compute a quantitative measure of similarity and most often generate a structural alignment. Taylor et al. [69] have compiled a comprehensive review describing challenges in protein structure comparison and its importance along with various proposed methods. The proposed methods differ from state-of-the-art techniques on the following two broad points: (1) choosing appropriate representation (2) algorithm for efficient and accurate retrieval of homologous structures from the database. The popular choices for representations include:

- Complete three dimensional coordinate information or partial coordinate information of backbone atoms,

- Representation of various elements using their properties such as ϕ-ψ angle, solvent accessibility, etc.

The first type of representations preserve sequential and topological relationships between individual elements in the structure. The methods developed to compare the first type of representations are partitioned into the ones using dynamic programming (DP) [3; 68] and the others not using DP [23; 32]. These methods are computationally expensive and do not scale well while comparing a large number of structures. Moreover, a large number of these comparisons do not yield satisfactory results since the structures are not related. To overcome these problems, researchers have proposed a two stage approach widely known as *filter and match paradigm*. The first stage of this approach employs a very fast filtering algorithm to obtain a small set of proteins which are very likely to be similar. These proteins are subjected to rigorous and computationally expensive structure alignment methods in the second stage, which is known as the match step. These methods achieve the desired speed in the filtering stage by representing proteins as vectors and comparing them in the space spanned by appropriate descriptors or features. For instance, the method proposed by Choi et. al. [26] represents protein structures using corresponding distance matrix. Rogen and Fein represented proteins with topological features of backbone using knot invariants [47]. Zotenko and co-workers represent each protein structure as a vector of frequencies of structural models, each of which is a spatial arrangement of triplets of secondary structure elements [17]. Several other feature based structure representation and comparison methods have also been proposed in the literature, such as Friedberg et al. [24], Tung et al. [12], etc. For a complete survey on these methods, refer to a survey by [69].

FragBag is the state-of-art fragment based structure comparison method proposed by researchers [25], where they use an interesting vector space representation for protein structures using fragments as the bases. Vector space representation of protein structure and retrieval using cosine similarity will capture the

similarity based on identity alone. Motivated by the success of topic models that capture similarity between documents at an abstract "topic" level, we propose an LDA based protein structure representation. We found that FragBag and LDA based representation outperforms each other for different retrieval requirements. In order to handle different requirements effectively, we propose a multi-view based retrieval framework that exploits multiple redundant representations of the protein structure.

2.3 Multiview Learning

Many real-world datasets possess data samples characterized using multiple views, e.g., web-pages can be described using both textual content in each page and the hyperlink structure between them. It has been shown that the error rate on unseen test samples can be upper bounded by the disagreement between the classification decisions obtained from independent views of the data [54]. This relatively new machine learning technique, commonly called as multiview learning [6; 31], has been predominantly successfully used in conjunction with semi-supervised and also unsupervised approaches [6; 31; 65; 15; 2; 52; 35; 55; 56; 13].

In many domains class labels are expensive to obtain and hence scarce, whereas unlabeled data are often cheap and abundantly available. Fortunately, many datasets are naturally partially labeled. For example, in the task of web page classification, it is possible to get partial labels on web pages using social bookmarking sites, open directory project, etc. Moreover, unlabeled data can be used to minimize the misclassification rate by enforcing consistency between the classication decisions based on different views of the unlabeled examples. In this work [1], Blum and Mitchell proposed an iterative, alternating co-training method, which bootstraps a set of classifiers from high confidence labels. It works by repeatedly adding pseudo-labeled unlabeled examples into the pool of labeled examples and retrains the classiers for each view. This process is repeated until the convergence

criteria is satisfied.

Co-EM is a related multi-view learning algorithm [31], which extends the co-training algorithm to operate simultaneously on all unlabeled samples in an iterative batch mode. Co-EM was used with SVMs as base classiers [62], and subsequently in unsupervised learning [53]. However, co-EM is not really an EM algorithm since it lacks a clearly dened overall log-likelihood that monotonically improves across iterations. It also suffers from local maxima problems. Recently, some co-training algorithms jointly optimize an objective function which includes loss terms for classiers from each view, and a regularization term that penalizes disagreement between the classication decisions based on different views.

This co-regularization approach holds the key intuition behind multiview learning. Krishnapuram et al. [8] proposed a co-regularization based approach for multi-view learning. It simultaneously learns multiple classiers by maximizing an objective function that penalizes misclassifications by individual classifiers, and also includes a regularization term which penalizes disagreement between different views. This co-regularization framework improves upon the baseline co-training and co-EM algorithms. The co-regularization approach was later adapted to two-view spectral clustering [64] and pair-wise clustering algorithm [15] which aims at nding pairwise clusters with multiview observations. This approach was subsequently adopted for semi-supervised classification and regression based on the reproducing kernel Hilbert space (RKHS) [65; 63; 67].

In the seminal work in this field [6], Blum and Mitchell proposed a co-training based multi-view learning algorithm which bootstraps a set of classifiers from high confidence labels. This paper provides PAC-style guarantees if (a) there exist weakly useful classiers on each view of the data, and (b) the views are conditionally independent given the class label. It could be impractical to expect such natural splits in all the domains. So, researchers started working towards finding relaxations and necessary conditions for multiview learning to be effective. Nigam and Ghani [31] presented other SSL algorithms for multi-view learning such

as co-EM, self-training. They also analysed the conditions necessary for co-training to be successful. Following Nigam and Ghani, many approaches for multi-view learning [65; 52; 35; 55; 56] exploit multiple redundant views to effectively learn a set of classifiers defined using each of the views. Balcan et al. [34] showed that co-training would be useful even if (a) there exist low error rate classifiers on each view, (b) these classifiers do not make misclassification when they are condent about their decisions, and (c) there would be cases where classifier on one view makes condent predictions while the classifier on the other view does not make confident predictions. Further, researchers found that multiview learning is advantageous compared to single view learning especially when the weaknesses of one view complement the strengths of the other [55; 13]. This forms the basic motivation for our proposed multiview based methods for collective classification on partially labeled graphs and similar protein structures retrieval tasks. In both the tasks, we exploit redundant views to improve the performance.

2.4 Summary

In the first part of this work, we deal with collective learning on partially labeled graphs. We treat content and link as two different views, where link is a weaker view. We propose multi-view learning based framework for collective learning that uses *content* and *link* views.

In the second part of this work, we deal with similar protein structures retrieval. As mentioned in the previous section, it is important to retrieve proteins whether they are close or intermediate or remote homologs. A single representation might perform well for one type of retrieval requirement. In order to meet diverse objectives, we propose a multi-view based framework that exploits multiple representations.

CHAPTER 3

Multi-view based Collective Learning

In this chapter, we explain the proposed multi-view based approaches for collective learning. Typically, collective classification algorithms use a flat feature vector by combining the content and link information. Motivated by the success of methods which leverage upon multiple views independently[2; 6; 13; 15; 31; 35; 52; 55; 56; 65], we study the impact of handling the content and link information separately as two different views. This problem differs from typical multi-view learning[6] because the content view, by labeling data points, not only provides labels for the link view but also alters the link view. In the first part of this work, we use a multi-view based approach for sentiment analysis. We refer to the proposed framework as MGSA (explained in the following sections). We then discuss the limitations of MGSA. An attempt to overcome some of the limitations identified with the framework involves a two step multi-view learning based procedure for collective classification (MVCL). We use the standard benchmark datasets to show the effectiveness of the proposed framework. There is immense promise in working on approaches aimed at addressing the other limitations of the sentiment analysis framework and adapting multi-view learning based framework for sentiment analysis. In the next section, we elaborate the intuition behind the proposed approach.

3.1 Overview

In this section, we explain the motivation and intuition behind the proposed approach. The scenario can be formally stated as follows: we are given a network $G = (O; L)$, where O is a set of objects and $L \subseteq O \times O$ is a set of links (edges). We refer

to network data as network, its nodes as objects and edges as links throughout this chapter. The network objects are partially labeled, with only a small fraction of them being labeled with labels from a finite set of categories C. For example, one can consider a network of people in a social networking site, where the links are between people who are friends of each other. Each person's background belongs to one of the following categories: engineering, finance or medicine. The background of some people are known, and the task is to categorize the background of remaining people. Each person can be classified based on her own attributes such as conversations data, groups enrolled, etc. In addition to this, the friend's information (link information) is an useful data that can be leveraged for increasing the classification performance. We use only the class information from linked objects, since it has been shown that using neighbors' attributes reduces the performance[48; 59]. We refer to class distribution and other derived statistics of neighboring objects' labels as link information throughout the text. In this section, we develop the intuition behind the algorithm for a network data with two views (content and link features). It can be extended to cases where there are multiple representations of content data and multiple link types between objects. For example, in a citation network the *publication* object can have *title*, *abstract* and *paper content* as content features, and the links can be *citations* and *author based* links i.e., if two papers have a common author, then they will be linked.

Most of the collective classification approaches, train a base classifier f using the labeled network data L, and apply it iteratively on previously unseen partially labeled network. The base classifier is trained using the features of objects. It is a combination of objects' attributes and link information. These approaches have the following drawbacks

- It might not be possible to have enough labeled data to train a base classifier in all the scenarios.
- Collective classification techniques do not leverage the unlabeled data during training phase. It has been shown that use of limited labeled data and vast unlabeled data which is easily available, improves the performance of the trained classifier[49; 70].

In order to overcome these drawbacks, we propose a multi-view learning based algorithm in a semi-supervised setup which utilizes both content and link views effectively to learn from the unlabeled data also. It performs well even in cases with little labeled data. Let us assume that a base classifier has to be trained on the given network shown in Figure 3.1(a). Feature vector can either be the combined flat vector, or the structured representation where content and link information are maintained separately. The link information is incomplete for the given network, i.e., none of the neighbors of the labeled datapoints are labeled. So, the link information part of the feature vector would have zero values. Hence, it can be seen that the classifier on the combined vector would end up using the content information only. If the link information is partially complete, as shown in Figure 3.1(b), then E and F nodes are complete to be used in training a base classifier. The usefulness of D depends on the class distribution of its neighbors, i.e., if E and F belong to the same class, then classification of B to any arbitrary class would not influence the class of D. Else, the node D might turn out to be a noisy input for the base classifier. It must be noted that it would not be possible to obtain complete link view in all the cases.

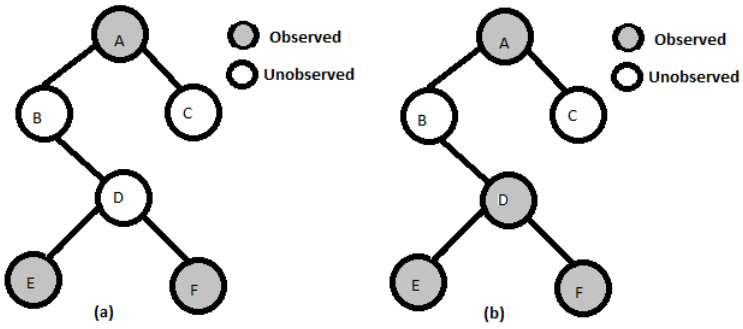

Figure 3.1: Sample network data

The collective classification algorithms usually involve a one-step bootstrapping procedure to initialize the labels of unlabeled nodes. It is done by training a base classifier on the labeled nodes and using it to predict the labels of unlabeled nodes. We use this one-step bootstrapping procedure to bootstrap labels for

sentiment analysis task (MGSA). The bootstrapping procedure not only initializes the unlabeled datapoints, but also bootstraps the link view which will be used by the iterative learning procedure. So the error induced by the first step not only affects the quality of labels but also the quality of link view. In the next step, the classifier would not only be training using noisy labels, but also on noisy link features. Ideally, the bootstrapping procedure must use the available content and link information effectively, i.e., confident predictions using content and link views must be used to label the unlabeled data. From the example shown in Figure 3.1(a) it can be seen that each of the representations will have confident predictions of objects' label. If nodes "E" and "F" belong to the same class, then node "D" can be confidently labeled using its link information. Similarly node "C" can be labeled using "A". The confident labels provided by link information can be considered as truth to train the attribute based classifier. And, the confident predictions using the attribute based classifier, can aid the link based label propagation. This must be repeated until all the datapoints are labeled. Though it is not possible to train a global classifier on link data, it is possible to use a local link based label propagation technique, along with a content classifier to bootstrap the labels. We refer to this as *Multi − view based bootstrapping procedure* in MVCL.

The bootstrapping step is followed by an iterative learning procedure where a classifier is learnt on content and link information. In MGSA, a classifier is learnt on content alone, and a simple voting is used for link information. It assumes homophily, which may not be the case in all domains. We overcome this limitation in MVCL by learning a classifier on link information also. We evaluate both MGSA and MVCL frameworks on standard linked datasets.

3.2 Sentiment Analysis

Sentiment analysis is the task of identifying the sentiment expressed in a given piece of text about the target entity discussed. Due to Web2.0, the number of

online reviews and users are increasing exponentially. Consequently, this field has gained huge importance in market analytics. Depending on the target entity, the granularity of the analysis varies. The target entity can be the product itself, for example "Canon digital camera", which is called as coarse-grained analysis. On the other hand the target entity can also be finer, capturing various features of a product, for example "clarity of the camera", which is called as fine-grained analysis. It can be achieved by performing analysis at various levels of granularity — document level, paragraph level/sentence level/phrase level — which basically captures product level, sub topic level or feature level target sentiments. The former refers to the physical structure of text taken for analysis, while the latter corresponds to the logical level. We use this notion of physical and logical levels in granularity throughout the thesis. There are models built at each level individually[9; 42; 44; 45; 46; 66] which are called as independent models[40]. Considering the nature of information available, the emphasis of the recent approaches is towards exploring the intra dependencies at each level and inter dependencies between the levels. Intuition behind intra dependency between entities at a single level is explained using the following example. In the piece of automobile review text given below, if the sentiment about manual gear shifter is unknown,

"The manual gear shifter is rubbery."

then the sentiment about similar features such as driving experience can help in disambiguating the opinion.

"..has an unpleasant driving experience.."

Inter dependency between coarser and finer levels are also useful to predict the unknown opinions[7; 40], for instance if sentiment of a document is known, then majority of the sentences should have same sentiment as the document and vice versa. This forms the basis of the proposed model, since we use these properties in the multi-grain collective classification framework. Since the above mentioned

approaches[7; 40] assume fully labeled corpus, which is not naturally available and huge amount of web data is partially labeled, we propose a multi-grain collective classification algorithm for the semi-supervised environment which is partially labeled at fine and coarse-grained levels. The target entities focussed in this work are document and sentence level.

3.2.1 Related Work

Pang and Lee[7], use the local dependencies between sentiment labels on sentences to classify sentences as subjective or not, and the top subjective sentences are used to predict the document level sentiment. It can be seen as a cascaded fine to coarse model and was shown to be better than other document level models. This work gives the basic motivation to study these dependencies. McDonald, Ryan et al.,[40] employ a joint structured model for sentence and document level sentiment analysis, which models document level sentiment using sentences level sentiment, sentence level sentiment using other sentences in the local context and document-level sentiment. This model was proven to be better than cascaded models (both fine to coarse and coarse to fine) and independent models at both the levels. Both the approaches assume to have fully labeled data, minimum cut based approach[7] uses labeled sentences to predict document level sentiment, and the structured model[40] uses labeled documents and sentences to build the joint model. Since it is known that getting fully labeled corpus at any of the levels is not possible, as the data available in web is naturally partially labeled, and it would involve human annotation to get fully labeled corpus, we do not assume the data to be fully labeled at any level. Secondly these approaches capture the structural cohesion between the sentences i.e., sentences occurring in physical proximity (next to each other) and not the logical cohesion, i.e., sentences discussing about similar features as mentioned in Introduction section. Dependency between sentences based on logical cohesion captured using anaphora and discourse relations has been shown to perform better than other approaches[58]. Also it is a sentence level approach

and not a multi-grain approach as the proposed model. The issues with the existing approaches and the way proposed model differs from those approaches is briefed below

- To our best knowledge, no framework has been proposed for multi-grain sentiment analysis in a semi-supervised environment, i.e., data is not fully labeled at any of the levels. Only a subset of documents are labeled with document level sentiment, and again only a subset of sentences in a document are labeled in the form of pros and cons.
- In the above mentioned approaches, dependencies captured either at structural or logical level expect the text to be written in an ideal manner for better performance. Only if the sentences discussing related features are written next to each other, the structural cohesion will work. And the logical cohesion captured using discourse graph[58], expects the sentences to explicitly have anaphoric and discourse relations.
- Another disadvantage of the discourse graph based approach is that it needs anaphora resolution and discourse structure identification to be performed for all input documents. It also ignores background domain knowledge and fully relies on discourse graph construction methods. It is obvious that some form of background domain knowledge will be available in the form of domain taxonomy, knowledge bases like Wordnet, etc. Also with the availability of huge amount of text it is possible to perform knowledge engineering and build a domain knowledge base, which captures features of a domain and similar features for the same.

In this work, we propose a collective classification algorithm which performs multi-grain sentiment classification in a semi-supervised environment. Intra-dependency at sentence level is captured using domain knowledge base, i.e., relation between features of a domain. The advantage is that it can be prebuilt, and instantiated for each document. This can be seen as adding domain knowledge to avoid sparsity that might arise in discourse graph based techniques, that capture logical cohesion between sentences. Since construction of domain knowledge is not the focus of this work, apart from briefly explaining the knowledge base used for this work in the experimental section we do not discuss various methods available or analyse them further.

3.3 Proposed Approach

Let C be a corpus of web based review documents. A subset of review documents have pros and cons section, which has positive and negative sentences discussing about different features of the target product. An example sentence is given below, taken from pros section of a laptop review from CNET website.

"Slim design; easy-to-use Intel Wireless Display built-in; speedy Core i5 processor."

The pros and cons sentences have phrases which are generally comma or semi-colon separated, that discuss about different features of the review's target product. Since not all the sentences in the document are labeled, those documents are referred to as sentence level partially labeled data S. Subset of documents contain overall sentiment label of the product, in the form of 5 stars, rating that scales between 1 and 10, or binary labels such as YES or NO. In this work, only binary labels are considered. Since only a subset of documents have overall sentiment label, they are referred to as document level partially labeled data G. Documents that either have pros and cons section or document level overall sentiment label or both, are referred to as multi-grain partially labeled data M. Documents that do not have any labels are called as unlabeled data U. So the web corpus is naturally partially labeled, $C{=}M \bigcup U$. In general, a document level sentiment label, can be seen as a function of sentence level sentiment labels. It is stated formally as follows, D denotes a document and a document contains a set of sentences, $D = \{s_1, s_2, s_3, ..., s_n\}$. Sentiment label is denoted by Ω, and the function is given by Y_d, this yields the below formulation

$\Omega(D) = Y_d(\Omega(D_s))$, where $D_s = \{\Omega(s_1), \Omega(s_2), \Omega(s_3), ..., \Omega(s_{N_s})\}$, N_s is the number of subjective sentences in D

Each sentence s_j in turn can be seen as a set of sentiment terms O_j, where $O_j \epsilon s_j$. Thus sentence level sentiment label, $\Omega(s_j)$ can be seen as a function, Y_{us} of sentiment

term level labels $\Omega(O_{sj})$. Sentiment terms are words that carry polarity and the most commonly used class of words are adjectives. We refer to this as ***unigram based classification***. It is stated as below

$$\Omega(s_j) = Y_{us}(\Omega(O_{sj})), \text{ where } O_{sj} = \{\Omega(O_1), \Omega(O_2), \Omega(O_3), ..., \Omega(O_{N_u(j)})\}, N_u(j) \text{ is the number of sentiment terms in sentence } s_j$$

The precision issues of unigram based classification approach are briefed as follows

- **Domain adaptation** : Classification of unigrams can either be based on general lexicon W, which has positive and negative terms, or using labeled data — S, G or M, depending on the granularity. Since the lexicon W, is not domain and context specific[46], there are cases where it would fail. This includes polarity mismatch of terms in the lexicon for different domains, and also expansion of the lexicon for evolving domain oriented sentiment terms. For example, it is not possible to predict the sentiment of the term *"rough"* in the sentence *"...surface is rough..."*, independent of the domain. It will be negative in the case of products like camera, and positive in the case of tyre. General lexicon might fail in this case depending on the domain. Also, we pose the missing sentiment terms as precision issue by adding the sentiment terms found in the corpus to the lexicon and assigning arbitrary labels to it. So polarity has to be relearnt using the evidences in multi-grain partially labeled dataset M.

- **Context specificity issues** : Set of sentiment terms have different opinions in different context, i.e., when they modify different target features. For example, the same sentiment term *"huge"* will be positive in *"huge win"*, and negative in *"huge loss"*. Since it is not possible to capture the context using the sentiment term alone, the unigram based classification faces precision issues in these cases.

Alternatively, s_j can be seen as a collection of constituent target feature term and sentiment term pairs, thus sentiment label at sentence level $\Omega(s_j)$ can be seen as a function, Y_{ts} of tuple's labels ($\Omega(T_j)$). We refer to it as ***tuple based sentence classification***. It is given formally as below

$$\Omega(s_j) = Y_{ts}(\Omega(T_j)), \text{ where } T_j = \{\Omega(F_1, O_1), \Omega(F_2, O_2), \Omega(F_3, O_3), ..., \Omega(F_{N_t(j)}, O_{N_t(j)})\}, N_t(j) \text{ is the number of tuples in sentence } s_j$$

Inference based on *tuple based sentence classification* would give high precision but low recall[44]. We denote the lexicon with labeled unigrams as L_u and bigrams as L_t. Rather than relying on L_u or L_t alone to infer the unknown label of a tuple (F_q, O_q), we infer it based on L_t and L_u using a backoff model. The intuition is that L_u can be used to infer sentiment of a tuple, when it cannot be inferred using L_t. It is obvious that the label identified using L_t is more reliable than L_u due to domain adaptation and context specificity issues faced by unigram based sentence classification technique. Thus we assign a confidence flag for both the kinds of prediction. Predictions using L_t are assigned *High* confidence, whereas those using L_u are assigned *Low* confidence. The Backoff inference procedure is given in Algorithm 1. The iterative learning procedure is given Algorithm 2. In this work, the functions Y_d, Y_{us} and Y_{ts} are just simple voting.

3.3.1 MGSA : Multi-grain Sentiment Analysis Framework

The notations for the iterative procedure are re-established and simplified as follows

l - Sentiment label space, which is $\{+, -\}$

$N_s(i)$ - Number of subjective sentences in the document D_i

$N_s(il)$ - Number of subjective sentences with label l in D_i

$N_u(j)$ - Number of unigram sentiment terms in the subjective sentence s_j

$N_u(jl)$ - Number of unigram sentiment terms in the subjective sentence s_j with label l

$N_t(j)$ - Number of tuples in the subjective sentence s_j

$N_t(jl)$ - Number of tuples in the subjective sentence s_j with label l

K_b - Neighborhood structure of feature terms given by the domain knowledge base. For example, it can contain the information that engine and performance of an automative are related. Thus the sentiment of related feature terms will be correlated in a given review.

L_u - Lexicon with labeled sentiment terms from lexicon W, which are relearnt for the domain using M, thus L_u is initialized with class labels. L_u has the structure with 5 elements $[O, Flag, Label, c_+, c_-]$, where O denotes the sentiment term, Flag has commit and not-commit status which means whether the class label is committed or not. Label gives

the class label, c_+ and c_- denote the count of occurrence of tuple in positive and negative contexts during the iteration i.e., number of times classified as positive and negative respectively. The querying that was mentioned in backoff model, $L_{lu}(O)$ which denotes the label returned using L_u for the sentiment term O is done as follows.

$$If(Flag == commit) \rightarrow Return\ Label$$
$$Elseif(Flag == uncommit) \rightarrow Return\ argmax_l(c_l)$$

L_t - Lexicon with labeled tuples initialized using pros and cons section in M. L_t has the structure with 6 elements $[F, O, Flag, Label, c_+, c_-]$, where F denotes the feature term, and other 5 tuples are same as in L_u. The querying procedure in backoff model and the way counts are updated during iterative procedure is same as what is done for L_u.

$\Omega(t)$ - Sentiment label of a tuple

$(\Omega(t), Confidence)$ - Pair denotes sentiment label of a tuple and confidence of inference given by backoff inference procedure

$\Omega(s)$ - Sentiment label of a sentence, which was defined as the function of labels of constituent tuples. The function used in this work is $maxlabel(x)$, where x is a collection of labels. $maxlabel(x)$ returns the label which occurs predominantly in the input x, i.e., a sentence is labeled with the maximum occurring label of its tuples, which is given below

$$\Omega(s_j) = maxlabel(\Omega(T_j)), \text{ where } T_j = \{\Omega(F_1, O_1), \Omega(F_2, O_2), \Omega(F_3, O_3), ..., \Omega(F_{N_t(j)}, O_{N_t(j)})\},$$
$$\Omega(T_j) \epsilon l.$$

Note that any function can be used in place of max, for instance a linear classification model. Since the focus of this task is to show how the proposed model improves the accuracy of any classical learning algorithm and not to propose a new fine-grained or coarse-grained classifier, the simplest classification function is chosen.

$\Omega(D)$ - Sentiment label of document given by bottom up label propagation of the constituent sentences. Similarly here also *maxlabel* is chosen as the function thus giving the

following formulation

$$\Omega(D) = maxlabel(\Omega(D_s)), \text{ where } D_s = \{\Omega(s_1), \Omega(s_2), \Omega(s_3), ..., \Omega(s_{N_s})\}$$

γ_{tli} - Number of neighbor tuples for a tuple t with the opinion l in a document D_i

Example scenario for two documents D_1 and D_2 is given in Figure 3.2. As we mentioned above, the dataset is naturally semi-supervised both at document and sentence level. We can assume that a subset of nodes are observed and the other nodes are unobserved. We do not explain the example much since the idea is to show the graph structure. Multi-grain iterative classification is a joint model that helps in predicting the labels of unobserved nodes to get completely labeled dataset, and also acquire more evidence for unigrams and tuples to update the lexicon L_u and L_t.

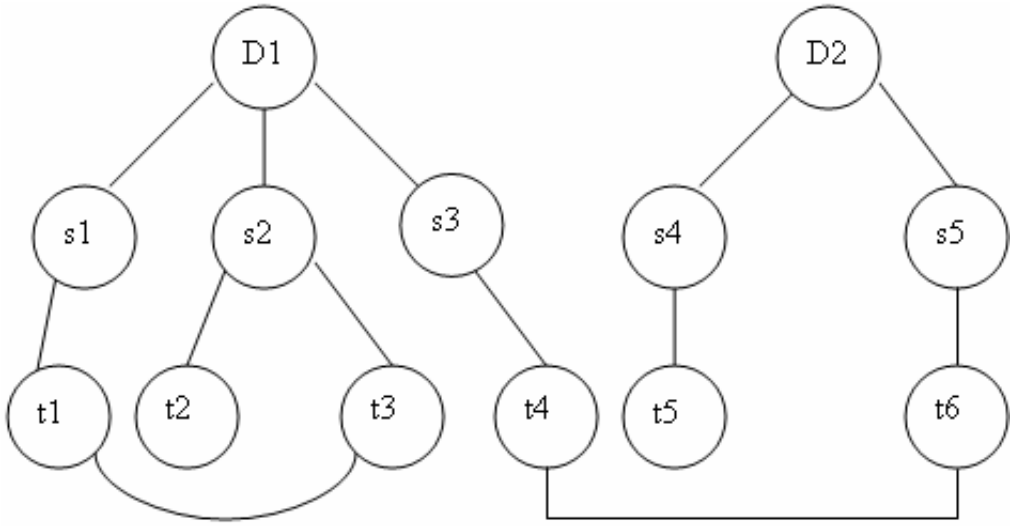

Figure 3.2: Example showing neighborhood for two documents

Initialization : L_u and L_t are initialized. Only subjective sentences in a document are taken for analysis, and presence of adjectives is taken as the indicator for subjectivity analysis[5]. Since the iterative procedure is shown to have same

performance for any arbitrary ordering of nodes[27; 48], we choose an arbitrary ordering for document nodes during iteration and natural order of occurrence within a document for sentences. Iterative procedure is given in Algorithm 2. This process is repeated until it converges or stabilises. This occurs when there is little or no change between successive sets of labels assigned. The algorithm converges within 10-15 iterations for the datasets used in this work. The proposed algorithm can be seen as a cautious iterative relaxation labeling procedure [59]. It is a local method for learning on Markov Random Fields. Similar methods have been shown to converge faster on almost all the popular datasets, and do not seem to have any convergence issues empirically[27; 48; 49].

Algorithm 1 Backoff Inference Procedure(F_q,O_q)

Input: L_t, L_u, K_b
Output: Label of (F_q,O_q)

```
Label ← UNK
Label_l ← 0 //denotes count of neighbors with label, l
Confidence ← Low
if (F_q, O_q) in L_t then
    Label ← L_lt((F_q, O_q))
    Confidence ← High
else
    F_k ← Neighbors(F_q,K_b)
    for (F_k,O_k) in L_t do
        if O_q==O_k then
            Label_l ← Label_l + L_lt((F_k, O_k))
        end if
    end for
    Label ← argmax_l(Label_l)
    Confidence ← High
end if
if Label == UNK then
    Label ← argmax_l(L_lu(O_q))
end if
Return Label, Confidence
```

Algorithm 2 Iterative procedure

repeat
  for Document D_i in C **do**
    for Sentence s_j in D_i **do**
      for Tuple t_k in s_j **do**
        $(\Omega(t_k), Confidence) \leftarrow$ Backoff Model(t_k)
        if $Confidence == High$ **then**
          Commit labels, $L_t \leftarrow L_t \cup (t_k, \Omega(t_k))$
        else
          Update counts in L_u, L_t
        end if
      end for
    end for
  end for
  Update counts in L_u, L_t
  Update sentence and document sentiments
  Commit the current labels in L_u, L_t
until Convergence

3.4 Experimental Results

The review articles are taken from websites containing Automobile reviews. The websites used for this work include CNET, Epinions and Edmunds. CNET and Epinions articles have both pros & cons section and document level label. Edmunds articles contain pros & cons section only and no document level label. 100 articles were chosen arbitrarily from each website, thus forming a testset of 300 articles. The class distribution of document labels is — 120 positive and 80 negative documents among 200 documents which have document level label, and the remaining 100 have unknown labels. Note that all documents have binary labels: Yes or No, positive or negative. Lexical classifier is used as the baseline approach at both document and sentence levels. It uses count of positive and negative adjectives to classify. It must be noted that the collective classification framework uses lexical classifier as the base classifier. Experimental results are given in Table 3.1 and 3.2. P, R and F1 denote Precision, Recall and F1 measure respectively.

Class	P	R	F1	Method
+	0.381	0.334	0.355	Lexical classifier
-	0.278	0.245	0.260	Lexical classifier
+	0.73	0.61	0.664	Proposed model
-	0.63	0.56	0.593	Proposed model

Table 3.1: MGSA Performance : Sentence level classification

Class	P	R	F1	Method
+	0.55	0.39	0.456	Lexical classifier
-	0.225	0.30	0.257	Lexical classifier
+	0.78	0.734	0.756	Proposed model
-	0.56	0.498	0.527	Proposed model

Table 3.2: MGSA Performance : Document level classification

3.5 Summary and Shortcomings

The following are the observations regarding the proposed framework for sentiment analysis:

- We perform sentiment analysis in a semi-supervised environment using both content and link information. Unlike other techniques which use collective classification for sentiment analysis, we do not assume to have fully labeled data[7; 40; 58]. Since we retain the structure of data, and use content and link information separately, we call this as a multi-view based approach to perform sentiment analysis.
- The results of MGSA outperform the baseline classifier. It proves that it can improve any local classifier's performance when used in this framework. Though the final results obtained are on par with state-of-the-art methods[42; 50], it has to be noted that this method is not a fully labeled approach and uses only a small amount of labeled data at sentence level.
- The main contribution of the proposed model, is the use of neighborhood structure of tuples within and between documents given by the domain knowledge base, by which the accuracy of any ordinary classifier can be improved. The classifier chosen in this work is lexical classifier which classifies a sentence using the most frequent label of its tuples. The cases where sentence level classification failed in the proposed model are those that needed other contextual evidences to be taken care. An example case is given below,

"The question is whether the car itself is as good as the wrapper it comes in"

So it indicates that with a better classifier (state-of-the-art) in place, the additional strength provided by the framework must help in better performance.

- Secondly, other main reason for mis-classification of sentences is wrong label initialization. Wrong labels induced by the bootstrapping step will be propagated further in the following iterative learning step. This is also known as *snowball effect*.
- The proposed multi-view based sentiment classifier learning algorithm, learns a classifier on the content view only. And, uses voting for the link view. Classification based on link information can be improved by using a global classifier.

In order to overcome some of the key limitations of MGSA framework, we further propose a multi-view collective learning approach(MVCL). The bootstrapping step in MGSA is replaced by a cautious multi-view based bootstrapping procedure. A classifier is learnt on the link view also. Other limitations involving the local classifier are left for the future work. We use standard linked datasets to evaluate the effectiveness of MVCL.

3.6 MVCL: Multi-view Collective Learning

In this section, we explain the MVCL framework which involves two steps: multi-view based bootstrapping and iterative structured collective learning. Given a partially labeled network G, the labeled subset of G is denoted by G_l, and the unlabeled subset is denoted by G_u. The network objects can have two or more representations A_i, where $A_i \ \epsilon \ A$. The task is to label the unlabeled objects to one or more classes, that belong to a finite set of categories C, and also build the classification models F_i on each of the objects' feature representations A_i.

3.6.1 Multi-view based Bootstrapping Algorithm

In many cases, it might not be possible to get objects and all its neighbors labeled to build a link classifier. In order to leverage the link view completely, we propose a

multi-view based bootstrapping step, where the content and (partially) incomplete link view bootstrap the link view together, i.e., initializes the labels of unlabeled objects. During this process, the local label propagation technique, LP, not only bootstraps link view, but also co-trains the content based classifier. In this work, Nearest Neighbor classifier is used as LP, where the neighbors are given by links in the network. The algorithm is generic for multiple content and link types of an object. For simplicity, we explain it here for a case with single content and link type. Initially, a classifier is trained on the content view of labeled data, and the n most confident predictions are added to labeled set. On the other hand, LP not only uses neighbors' class labels, but also the ratio of labeled and unlabeled neighbors to commit the label of an object. The intuition behind this is, LP should commit the label of an object as x, only if majority of its neighbors belong to label x, and even if all the unlabeled neighbors turn out to be belonging to some arbitrary label y, the label prediction must not change. This is done to avoid noisy label propagation. Among these items, n most confident predictions are added to labeled set. If the prediction of content based classifier and LP disagree, then the object's label is not committed. This step is performed to avoid disagreement between multiple views. The bootstrapping procedure is given in Algorithm 3.

We provide an overview of Algorithm 3 here. State of the network objects is maintained as *OLD*, *NEW* and *UNK* : it denotes objects for which the label is given, objects labeled by the iterative bootstrapping step, and objects that are unlabeled. Step 1 and 2 indicate the initialization of the state of labeled and unlabeled objects. Step 2 deals with content views of the objects. In Step 5, a classifier is built on each of the content views. It is followed by Step 8, where the n most confident predictions are added to the labeled set. Step 15 checks for view disagreement : if the prediction of content based classifier and current label disagree, then the object's label is not committed. It could be either of the cases: LP labeled the object with different class label, or content classifier labeled it differently in previous iterations. Former case is view disagreement, and the latter case is change of prediction with

new evidence. The change due to latter case will be automatically corrected in the following iterations. From Step 22, the procedure involves the use of link based label propagation, LP. Step 23 is done to avoid noisy label propagation : LP should commit the label of an object as x, only if majority of its neighbors belong to label x, and even if all the unlabeled neighbors turn out to be belonging to some arbitrary label y, the label prediction must not change. The last few steps are explained after the algorithm.

$conf(LP(O))$ refers to the maximum value of ratio of number of neighboring objects belonging to any class to total number of neighbors for the object. It is given as $\max(\frac{Classcount(c)}{\#neighbors})$, $\forall c \in C$. $conf_{unk}(LP(O))$ refers to the ratio of number of unlabeled neighbors to total number of neighbors for the object, $\frac{Count(unk)}{\#neighbors}$. Among those objects which qualify this condition, in Step 28, n most confident predictions using LP are added to labeled set. In Step 35, the disagreement check similar to content view is performed for link view.

3.6.2 Multi-view based Iterative Learning Algorithm

After this bootstrapping step, we propose an iterative structured collective learning algorithm. Classifiers are learnt on content and link views separately, and the predictions are combined using ensemble operators[30]. The operators used in this work are Product and Max. The iterative algorithm is given in Algorithm 4.

Algorithm 3 Multi-view based Bootstrapping Algorithm

Input: G is a partially labeled network data, F_a is a set of classifiers for content representations of data, n is the number of objects to be committed in each iteration by the classifiers
Output: Bootstrapped network data, trained content data classifiers

Assign State of $O = OLD$, where $O \in G_l$
Assign State of $O = UNK$, where $O \in G_u$
repeat
  for Content feature representation $A_a \in A$ **do**
    Build classifier F_a on $A_a(O)$, where $O \in G_l$
    $k \leftarrow 1$
    repeat
      Pick O_k, the kth confidently labeled object
      if State(O_k) == UNK **then**
        $Label(O_k) \leftarrow F_a(A_a(O_k))$
        $G_l \leftarrow G_l \bigcup O_k$
        $k \leftarrow k + 1$
        Assign State of $O_k = NEW$
      else
        if $Label(O_k)! = F_a(A_a(O_k))$ AND State(O_k)==NEW **then**
          $G_l \leftarrow G_l \setminus O_k$
          Assign State of $O_k = UNK$
        end if
      end if
    until $k < n$
  end for
  for Link type $A_l \in A$ **do**
    confitems $\leftarrow$ max($conf(LP(O))) > conf_{unk}(LP(O))$
    Ω(confitems) $\leftarrow$ Count(confitems)
    $n_{link} \leftarrow$ min(Ω(confitems), n)
    $k \leftarrow 1$
    repeat
      Pick O_k, the kth confidently labeled object
      if State(O_k) == UNK **then**
        $Label(O_k) \leftarrow arg\,max_{label} conf(LP(O_k))$
        $G_l \leftarrow G_l \bigcup O_k$
        $k \leftarrow k + 1$
        Assign State of $O_k = NEW$
      else
        if $Label(O_k)! = arg\,max_{label} conf(LP(O_k))$ AND State(O_k)==NEW **then**
          $G_l \leftarrow G_l \setminus O_k$
          Assign State of $O_k = UNK$
        end if
      end if
    until $k < n_{link}$
  end for
until Unlabeled object exists or fixed number of iterations is reached
Return G, F_a

Algorithm 4 Iterative Structured Collective Learning

Input: Bootstrapped network data G, F is a set of classifiers F_i, where F_i is a classifier for the feature representation A_i, $A_i \, \epsilon \, A$

Output: Fully labeled network data, trained classification models F

repeat

Reestimate parameters of the classifier $F_i(A_i)$, using the representation A_i of labeled objects

Label(O) $\leftarrow$ Ensemble Operators($F_i(A_i)$), $O \, \epsilon \, G_u$

until Convergence or fixed number of iterations

Return G, F

3.7 Experimental Results

Citeseer, Cora, and WebKB link mining datasets are used for benchmarking the classification task on partially labeled network[48]. Details of the datasets are given below

- CiteSeer dataset consists of 3312 scientific publications and 4732 links. The task is to classify the publications object into one of six classes.
- Cora dataset consists of 2708 scientific publications and 5429 links. The task is to classify the publications object into one of seven classes.
- The WebKB dataset consists of 877 scientific publications and 1608 links totally, combining the four university datasets. The task is to classify the publications object into one of five classes.

The results of the bootstrapping step is given first. The multi-view based bootstrapping step is compared against the baseline approaches. The baseline approaches includes the following

- Build a classifier on content data of the labeled objects, and use it to predict the labels of unobserved objects
- Build a classifier on content and link data of the labeled objects appended, and bootstrap labels of unobserved objects using its prediction.

The results of the bootstrapping step is given first. The multi-view based bootstrapping step is compared against the baseline approaches. Comparison of

content only (Co), content and link appended (Co-Link), and multi-view based bootstrapping technique (MV) using Naive Bayesian (NB), and K-Nearest Neighbor (KNN) as the base classifiers is given in Table 3.3-3.5. In the case of Naive Bayesian classifier, normalized posterior is used for predicting the output class label. In KNN, for combined flat vector A_{flat}, and for content based feature vector A_a, cosine similarity is used to compute distance between the feature vectors. KNN with link information uses the structural neighbors to predict the class label : this is performed by classifying an object using the predominant class among its linked neighbors. It must be noted that the neighbors computed using cosine similarity is different from the structural neighbors. The precision (P), recall(R), F1 measure (F1) and Accuracy(Acc) are averaged across 5 splits for different train-test ratios. The training and test split is done using stratified random sampling. The training ratio is given as TR in the tables. It can be seen that MV performs better Co and Co-Link in all the datasets in terms of F1 and Accuracy.

After the bootstrapping step, the structured collective learning, as given in Algorithm 4, is performed. EM-Like algorithm proposed by Lu and Getoor[49] is the baseline approach. Lu and Getoor have shown that the EM-Like algorithm performs better than approaches which combine the representations to a single vector. The comparative results are given in Table 3.6-3.8. ISCL-Product, ISCL-Max refers to Iterative Structured Collective Learning algorithms that use Product, Max ensemble operators respectively. It can be seen that the proposed framework performs better than the EM-Like collective learning approach. The best K value for KNN algorithm, for neighborhood based on cosine distance, was found to be 15. The best n value in the bootstrapping was 5. We also observed that both the bootstrapping and learning procedure converges within 15-20 iterations.

In order to verify the improvements of MVCL over MGSA, we evaluate the performance of MGSA on the standard linked datasets. We found that MVCL outperforms MGSA and the results are given in Table 3.9. MGSA uses a content classifier (Co) based bootstrapping procedure. From the bootstrapping results

given in Tables 3.3 to 3.5, it can be seen that this bootstrapping does not perform as good as Co-Link and MV. This affects the following learning procedure also. In MGSA, the labels were bootstrapped using backoff procedure. If the label of a tuple is predicted using bigram lexicon then the label is committed. Else, it is smoothed using the neighbor's class label distribution. In this case, if 'confidence' of the content classifier is greater than 0.85 (empirically chosen), then the label is committed. Else, the label is smoothed using the prediction based on link information.

Cora												
Naive Bayesian												
	Co				Co-Link				MV			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	70.56	64.92	67.62	70.12	75.62	71.88	73.70	75.89	78.6	76.77	77.68	79.74
40	73.63	71.71	72.66	74.74	78.59	77.68	78.13	80	79.85	79.8	79.83	81.38
60	74.69	73.49	74.09	75.97	80.26	79.89	80.07	81.68	81.89	79.72	80.79	82.5
80	75.27	75.55	75.41	77.14	80.43	81.18	80.8	82.42	82.96	81.26	82.1	83.2
KNN												
	Co				Co-Link				MV			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	66.88	59.43	62.93	65.25	71.54	66.43	68.89	71.25	75.32	65.85	70.27	73.97
40	69.16	64.47	66.73	69.07	74.96	72	73.45	75.6	76.81	72.78	74.74	76.83
60	72.42	67.59	69.92	72.12	77.53	74.68	76.08	78.14	77.95	75.88	76.9	78.96
80	74.69	70.64	72.61	74.68	80.02	77.93	78.96	80.61	80.98	80.16	80.57	82.12

Table 3.3: Bootstrapping performance for different training ratios using NB and KNN Classifier on Cora dataset: averaged across 5 splits

3.8 Conclusion and Summary

Key observations w.r.t the results include the following :

- The proposed approaches provide greater improvement on Cora than Citeseer, and on Citeseer than WebKB. Homophily exhibited by the datasets follows the same order, Cora > Citeseer >> WebKB. Homophily is measured in terms of autocorrelation of labels belonging to related objects: it is high for Cora(0.88), and Citeseer (0.83), and very low for WebKB (0.30)[41]. Though we build a global classifier on the link information in the iterative learning

Citeseer												
Naive Bayesian												
	Co				Co-Link				MV			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	69.46	63.33	66.25	69.57	70.49	64.85	67.55	71.16	72.64	69.26	70.91	73.08
40	70.67	66.93	68.75	72.35	72.17	68.43	69.16	73.74	72.89	70.19	71.52	75.13
60	71.45	68.28	69.83	73.24	72.93	70.02	71.44	74.87	73.09	71.08	72.07	75.56
80	71.6	70.03	70.81	74.31	73.95	72.47	73.2	76.36	75.04	73.1	74.06	76.95
KNN												
	Co				Co-Link				MV			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	66.42	59.46	62.75	65.36	63.27	59.91	61.54	66.36	69.84	61.51	65.41	68.53
40	70.48	64.94	67.6	69.83	68.46	64.69	66.52	70.96	69.7	64.59	67.05	72.58
60	72.7	67.2	69.84	71.68	70.61	67.42	68.98	73.14	71.97	68.74	70.32	74.03
80	67.07	65.02	66.03	70.24	69.94	66.49	68.17	72.07	71.19	69.39	70.27	74.26

Table 3.4: Bootstrapping performance for different training ratios using NB and KNN Classifier on Citeseer dataset : averaged across 5 splits

WebKB												
Naive Bayesian												
	Co				Co-Link				MV			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	63.23	44.93	52.53	72.09	63.17	44.98	52.55	72.18	63.48	45.18	52.79	72.65
40	75.98	52.95	62.41	76.39	74.63	53.08	62.03	76.48	76.28	53.45	62.86	77.03
60	76.18	57.67	65.64	79.19	74.26	57.92	65.08	79.7	76.51	57.8	65.85	80.09
80	75.71	61.15	67.66	80.96	75.68	61.35	67.77	81.19	75.97	61.48	67.96	81.54
KNN												
	Co				Co-Link				MV			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	56.01	37.49	44.92	66.02	56.09	37.54	44.98	66.08	56.27	37.76	45.19	66.36
40	75.6	43.38	55.12	70.82	75.69	43.36	55.14	71.07	75.98	43.65	55.45	71.38
60	75.27	46.22	57.27	73.03	75.37	46.26	57.33	73.1	75.78	46.49	57.63	73.51
80	87.86	48.8	62.75	74.8	87.9	48.84	62.79	74.9	88.23	49.02	63.02	75.46

Table 3.5: Bootstrapping performance for different training ratios using NB and KNN Classifier on WebKB dataset : averaged across 5 splits

Cora												
Naive Bayesian												
	EM-Like				ISCL-Product				ISCL-Max			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	79.27	73.12	76.07	77.9	79.77	76.62	78.16	81.46	79.96	78.78	79.37	82.73
40	80.32	77.17	78.71	80.33	80.92	81.07	80.99	83.03	80.98	80.28	80.63	82.85
60	81.29	79.12	80.19	81.7	82.98	80.32	81.63	83.19	83.29	81.22	82.24	83.97
80	82.16	80.66	81.4	82.9	83.86	81.87	82.85	84.63	83.76	81.89	82.81	84.09
KNN												
	EM-Like				ISCL-Product				ISCL-Max			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	70.12	64.47	67.18	69.38	76.1	65.98	70.68	74.29	76.82	66.78	71.45	75.89
40	72.34	69.63	70.96	73.64	77.83	73.36	75.53	77.96	78.45	74.15	76.24	78.86
60	76.37	73.72	75.02	76.87	78.67	76.14	77.38	79.16	79.15	76.61	77.86	79.89
80	81.58	79.5	80.53	81.89	81.12	80.02	80.57	82.39	81.76	80.54	81.15	82.81

Table 3.6: Collective Learning performance for different training ratios using NB and KNN Classifier on Cora dataset : averaged across 5 splits

Citeseer												
Naive Bayesian												
	EM-Like				ISCL-Product				ISCL-Max			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	72.98	66.01	69.32	72.62	73.99	69.42	71.63	74.82	73.19	69.34	71.21	74.39
40	72.99	68.42	70.63	74.21	74.89	70.23	72.49	75.48	74.56	69.95	72.18	75.32
60	73.67	69.67	71.61	74.91	75.91	71.38	73.58	76.89	75.62	71.23	73.36	76.7
80	74.84	72.01	73.4	76.45	76.16	74.16	75.15	77.87	76.51	73.03	74.73	77.19
KNN												
	EM-Like				ISCL-Product				ISCL-Max			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	68.1	59.97	63.77	66.73	69.58	61.78	65.45	69.78	70.75	62.81	66.54	70.37
40	68.48	63.99	66.16	70.68	70.35	64.57	67.34	73.1	70.57	64.88	67.61	73.51
60	70.87	66.34	68.53	72.93	73.45	68.78	71.04	74.91	73.12	69.61	71.32	75.11
80	73.09	66.96	69.89	73.06	73.04	69.56	71.26	75.14	73.45	69.54	71.44	75.46

Table 3.7: Collective Learning performance for different training ratios using NB and KNN Classifier on Citeseer dataset : averaged across 5 splits

WebKB												
Naive Bayesian												
	EM-Like				ISCL-Product				ISCL-Max			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	63.18	43.63	51.61	70.24	64.21	45.87	53.51	73.61	63.89	45.75	53.32	73.26
40	77.81	51.87	62.25	75.99	76.87	53.91	63.37	77.64	76.35	53.15	62.67	77.34
60	78.51	57.67	66.5	79.24	77.25	58.51	66.59	80.89	76.95	58.25	66.31	80.59
80	80.09	62.62	70.28	82	80.56	62.89	70.64	82.27	76.35	61.79	68.3	81.87
KNN												
	EM-Like				ISCL-Product				ISCL-Max			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	56.76	37.82	45.39	66.53	56.78	38.16	45.64	66.86	57.18	38.67	46.14	67.08
40	76.21	43.83	55.65	71.45	76.67	44.16	56.04	71.87	77.12	44.34	56.31	72.03
60	76.05	46.29	57.55	73.6	76.67	47.23	58.45	74.17	76.98	47.56	58.8	74.45
80	88.91	48.93	63.12	75.90	89.34	49.71	63.88	75.96	89.78	50.14	64.34	76.18

Table 3.8: Collective Learning performance for different training ratios using NB and KNN Classifier on WebKB dataset : averaged across 5 splits

Naive Bayesian												
	Cora				Citeseer				WebKB			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	76.95	75.78	76.36	78.06	72.18	66.26	69.09	72.71	62.87	43.28	51.27	70.71
40	78.75	78.85	78.80	80.48	72.71	68.98	70.80	74.37	77.38	51.75	62.02	75.64
60	81.30	79.15	80.21	81.72	73.20	70.29	71.71	75.13	78.37	56.76	65.86	78.62
80	80.38	81.32	80.84	82.38	74.08	72.63	73.35	76.57	79.71	62.43	70.02	79.43
KNN												
	Cora				Citeseer				WebKB			
TR	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
20	70.32	64.37	67.21	69.3	65.49	58.82	61.98	66.05	56.58	37.66	45.22	66.25
40	72.25	70.02	71.12	73.53	68.72	64.04	66.3	70.41	76.10	43.78	55.58	71.39
60	76.69	74.72	75.70	77.74	70.52	67.57	69.01	72.63	75.98	46.25	57.5	73.46
80	81.45	79.25	80.33	82.07	71.91	67.37	69.57	73.23	88.78	49.79	63.80	75.86

Table 3.9: MGSA's collective Learning performance for different training ratios using NB and KNN Classifier: averaged across 5 splits

step which must capture the higher order dependencies (not just homophily), the bootstrap step uses a local label propagation technique which assumes homophily. The results are encouraging to investigate replacing the local label propagation technique by a global technique which captures higher order dependencies between the labels of related objects.

- In general, performance improvement is more for cases with lesser training data, which makes the proposed approaches applicable for sparsely labeled datasets.

In this work, we have proposed a multi-view learning based collective classification framework, which performs collective learning on partially labeled network. The proposed framework can handle objects with multiple feature representations, and the framework is easily extendable to any number of feature representations. We have also proposed a two step framework, where a multi-view bootstrapping procedure is used to bootstrap the link view in first step. It is followed by an iterative structured collective learning framework, which uses ensemble operators to combine the predictions based on classifiers built using content and link informations. MGSA framework is not easily extendable to multiple views. It would be useful to apply MVCL for multi-grain sentiment analysis task where we can capture multiple views from the content information, intra-dependency at the same level of granularity and inter-dependency between the levels.

CHAPTER 4

Multi-view based Similar Protein Structure Retrieval

Protein structure similarity can be captured by not only matching fragments in the protein structure, but similar fragments (not just identical fragments) must also be considered to help protein structure comparison. This is achieved by modeling the protein structure using Latent Dirichlet Allocation (LDA)[14], which maps the fragments to a topic space using their co-occurrence information. Protein structure comparison at topic space performs a soft matching by considering similar fragments too. In order to handle different retrieval requirements from close to remote homologs, we propose a retrieval framework which utilizes multiple representations : plain vector space representation [38] of fragments in protein structure and topic space representation using LDA. We also evaluate a multi-view model that uses multiple representations (combining vector space model and LDA's topic representation) built with different fragment libraries. We empirically prove that the proposed multi-view framework which utilizes multiple representations outperforms other state-of-the-art techniques across all the retrieval requirements.

4.1 Introduction

The success of structure comparison methods can be measured based on their effectiveness in detecting closely and remotely homologous proteins [69]. The closely homologous proteins have similar structures with relatively less insertions and deletions. On the other hand, remote homologs possess significantly different structures. The similarity in these cases can be inferred based on similarity of structural fragments. Fragment level protein structure comparison works well

in practice as demonstrated by several methods [11; 25; 36; 37]. The first fragment based comparison method was proposed by Remington and Mathews, which performs rigid body superposition of fixed length backbone fragments from individual proteins [11]. The rigid body superposition was later used by Zuker and Somorjai to define distance between backbone fragments while comparing them using dynamic programming [37]. The fragment based structure comparison was also used in identification and ranking of local features [36]. For further details, the readers are referred to an excellent review paper by Taylor et al.[69]. Recently researchers proposed an interesting vector space representation of protein structures using fragments as bases [25]. The method, FragBag, appears to perform the task of retrieving similar structures efficiently and is the state-of-the-art method in fragment based structure comparison. FragBag represents each structure as a *bag of fragments*, which is a basic model of retrieval used in the area of text mining. The success of FragBag opens up many interesting avenues, where powerful language modeling techniques proposed in information retrieval/statistical natural language processing can be adopted for representing protein structures. They can achieve better performance in terms of efficiency and accuracy in identifying structural homologs. This work focuses on two important problems in this context:

- Effective protein structure representation that captures inherent relationship between fragments and facilitates efficient comparison between the structures,
- Effective framework to address different retrieval requirements.

We propose a new representation for protein structures based on Latent Dirichlet Allocation (LDA)[14]. For retrieving close and intermediate homologs, the LDA representation works better. On the other hand, for retrieving remote homologs the naive vector space model (FragBag) performs well. Since retrieval requirements could be diverse, it is necessary to build a model that is efficient across different requirements. In this work, we propose multi-view based retrieval frameworks to meet the challenges.

4.2 Proposed Approach

As mentioned previously, the framework for protein structure comparison has two subproblems to be handled. In this section, we will elaborate the proposed framework to address these subproblems. These proposed techniques draw a huge motivation from statistical NLP.

4.2.1 Representation of proteins in topic space

The key point of the proposed approach is to represent proteins as probability distributions over latent topics. Note that the topic is an abstract concept and is represented as a multinomial distribution over fragments. Given this representation, a collection of protein structures can be modeled using three-level hierarchical Bayesian generative model known as Latent Dirichlet Allocation (LDA)[14]. Intuitively, this formalism clusters similar fragments into topics, which provides significant advantage over models that perform fragment to fragment comparison (except identity) while comparing protein structures. We explain this concept with a simple example. Suppose we are interested in comparing two documents, one containing words **dog** and **cat** and the other containing **bark** and **mews**. Naive word level comparison of the two documents reveal that they are unrelated, when they actually talk about semantically related topics (dog-barking and cat-mews in this case). This example can be extended to protein structures, where fragments are entities equivalent to words in the document. The fragments are grouped into a topic in a probabilistic manner and the search for homologous proteins can be performed more accurately in the topic space. Before introducing formal aspects of the problem formulation, we describe the key ingredients:

1. A *fragment* f_i is the basic unit of protein structure. It is part of the fragment library of choice F. F=$\{f_1, f_2, \ldots, f_L\}$, where L is the size of fragment library F.
2. A *Protein* is a sequence of n fragments, denoted by $S = \{f_i | f_i \ \epsilon \ F\}$. The protein structure is converted into a sequence of fragments using the method described in [25].

3. A *Universe* is a collection of N proteins, denoted by U={s_1, s_2, . . . ,s_N}.

The graphical model representation of LDA is provided in Figure 4.1. It models the protein structure collection according to the following generative process:

1. Pick a multinomial distribution φ_z for each topic z from a Dirichlet distribution with parameter β.
2. For each protein s, pick a multinomial distribution θ_s from a Dirichlet distribution with parameter α.
3. For each fragment f_i in protein structure s, pick a topic z ϵ {1,. . . , K} with parameter θ_s.
4. Pick fragment f_i from the multinomial distribution φ_z.

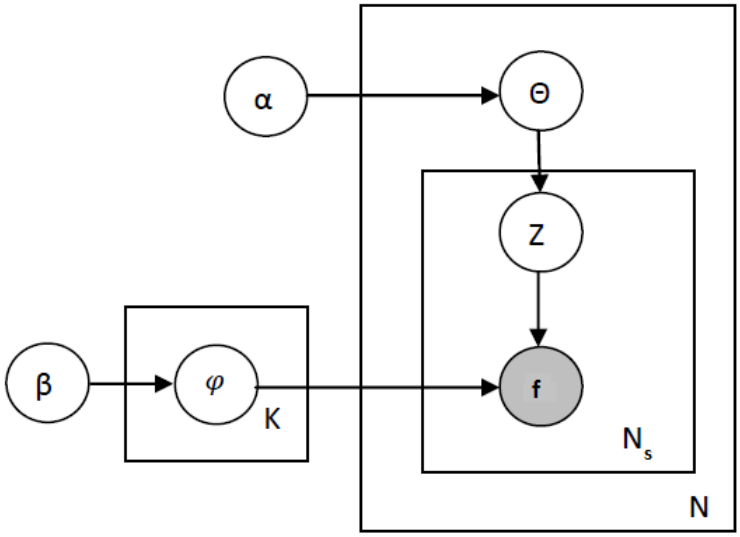

Figure 4.1: Graphical representation of LDA; K is the number of topics; N is the number of protein structures; N_s is the number of fragments in protein structure s.

According to the model, each protein is a mixture of latent variables z (referred to as clusters/topics), and each latent variable z_i is a probability distribution over fragments. Given N proteins, K topics, L unique fragments in the collection, we can represent $p(f|z)$ for the fragment f, with a set of K multinomial distributions φ over the L fragments, $P(f|z = j) = \varphi_f^{(j)}$. $P(z)$ is modeled with a set of N multinomial

distributions θ over K topics. One way to achieve this is to use Expectation Maximization to find the estimates of φ, and θ. It suffers from local maxima issues, and its hard to model an unseen protein since it does not assume anything about θ. LDA overcomes these issues by assuming a prior distribution on θ and φ to provide a complete generative model. It uses Dirichlet distribution [1] for choosing priors α for θ and β for φ.

The likelihood of generating a universe of protein structure collections is $P(s_1, s_2, \ldots, s_N) =$

$$\int\int \prod_{z=1}^{K} P(\varphi_z|\beta) \prod_{s=1}^{N} P(\theta_s|\alpha) (\prod_{i=1}^{N_p} \sum_{z_i=1}^{K} P(z_i|\theta) P(f_i|z, \varphi))\, d\theta\, d\varphi$$

Exact inference is LDA model is intractable and hence a number of approximate inference techniques such as variational methods[14], expectation propagation [61], and Gibbs sampling [20; 61] have been proposed in literature. We use Gibbs sampling based inferencing to estimate φ and θ. From a sample, $\hat{\varphi}$ and $\hat{\theta}$ are approximated using the following equations 4.1 and 4.2 after a fixed number of iterations, which is commonly known as burn in period.

$$\hat{\varphi} \approx (n_{i,j}^{(w_i)} + \beta_{w_i}) / \sum_{v=1}^{V} (n_{i,j}^{(v)} + \beta_v) \tag{4.1}$$

$$\hat{\theta} \approx (n_{i,j}^{(s_i)} + \alpha_{z_i}) / \sum_{t=1}^{T} (n_{i,j}^{(s_i)} + \alpha_t) \tag{4.2}$$

Here, $n_{i,j}$ is the number of instances of fragment f_i, assigned to topic $z = j$. α and β are hyper-parameters that determine the smoothness of the distribution. $n_{i,j}^{(s_i)}$ is the number of fragments in protein s_i that belong to topic $z = j$. Thus, the total number of fragments assigned to topic $z = j$ is given by $\sum_{v=1}^{V} n_{i,j}^{(v)}$. The total number

[1] Dirichlet prior is a conjugate prior of multinomial distribution

of fragments in protein s_i is given by $\sum_{t=1}^{T} n_{i,j}^{(s_i)}$. The terms, $\sum_{v=1}^{V} n_{i,j}^{(v)}$ and $\sum_{t=1}^{T} n_{i,j}^{(s_i)}$, are normalizing factors.

The work flow for building topic model is as follows:

1. We take collection of protein structures as an input. We process each structure and obtain the corresponding fragment by matching its substructures with the library. At the end of this process, we obtain a bag of fragments for each protein. This process is depicted in Figure 4.2.
2. We learn the topic model on the collection using the machinery described earlier in this section.
3. Each protein is then represented as a probability distribution over the latent topics discovered by LDA.

4.2.2 Multi-view based Retrieval

A simple framework for protein retrieval is given in Figure 4.3, where the universe of proteins are modeled using the representation R chosen. In order to rank the proteins based on the structural similarity for a query protein, the query protein is modeled and transformed to the same representation space R. Once the transformation is done, the protein structures in the collection are ranked based on their structural similarity with the query protein using a retrieval technique. Most simplest technique would involve a boolean vector representation for each protein. Here, the fragments from the fragment library are matched against the protein structure, and a vector of size of the fragment library is built. The vector has 1 in the position of fragments that are present and 0 in the place of fragments that are absent. And retrieval can be based on Jaccard's Coefficient [38]. It can be replaced by other IR techniques such as term frequency (TF), term frequency-inverse document frequency (TF-IDF), etc. The similarity metrics must be chosen according to the choice of representation [38]. We refer to this family of techniques as naive vector space models.

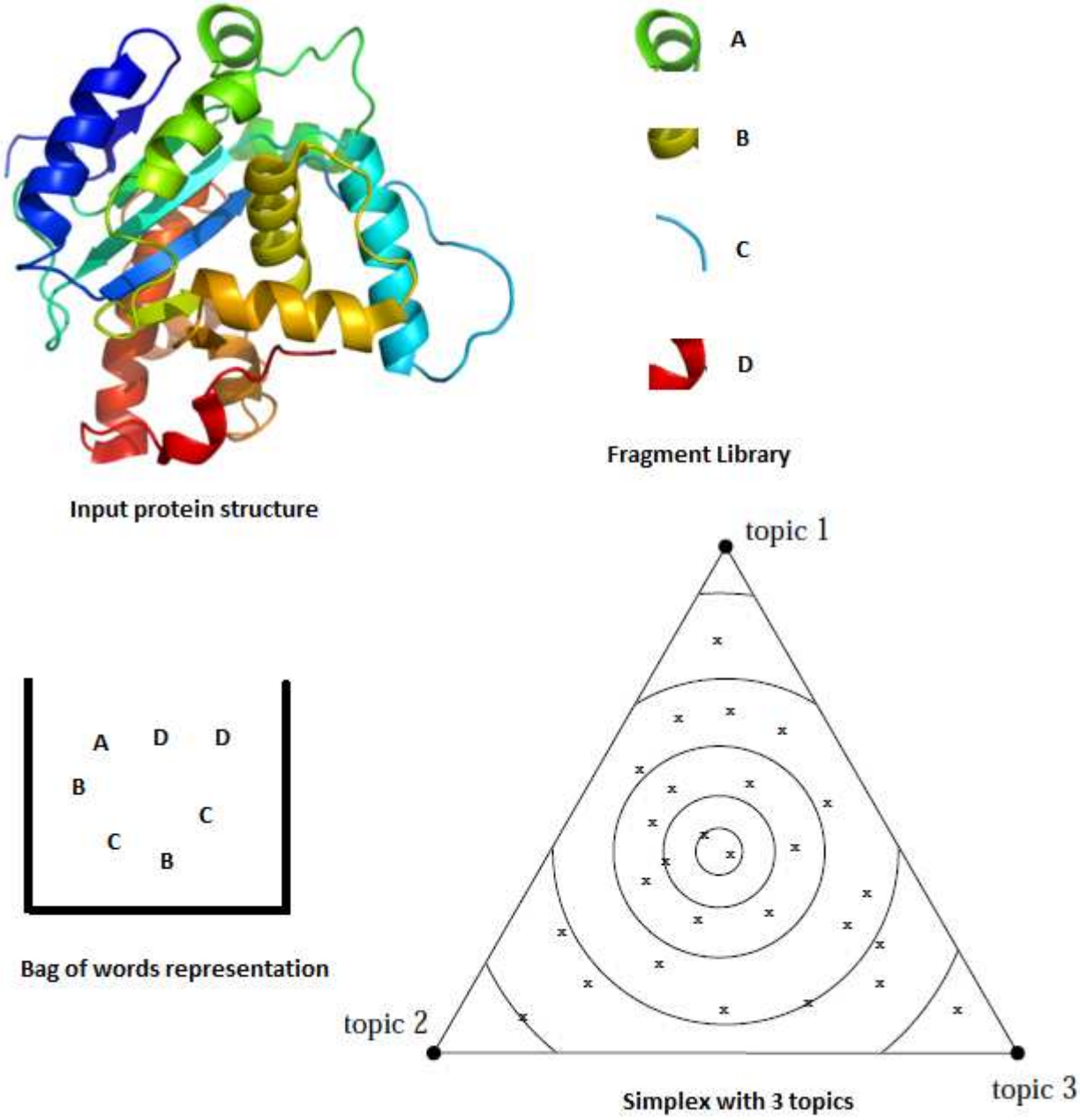

Figure 4.2: Example protein structure with bag of fragments and topic space representations; built for a given fragment library. (a) shows an example protein structure and (b) shows a given fragment library. Each substructure in protein is compared against the fragment library and the closest matching fragment is used to represent the substructure. Thus, we obtain bag of fragments representation for protein structure as shown in (c). We model the structure as a probability distribution over latent topics. In (d) we have shown a toy representation using three topics, which forms a simplex.

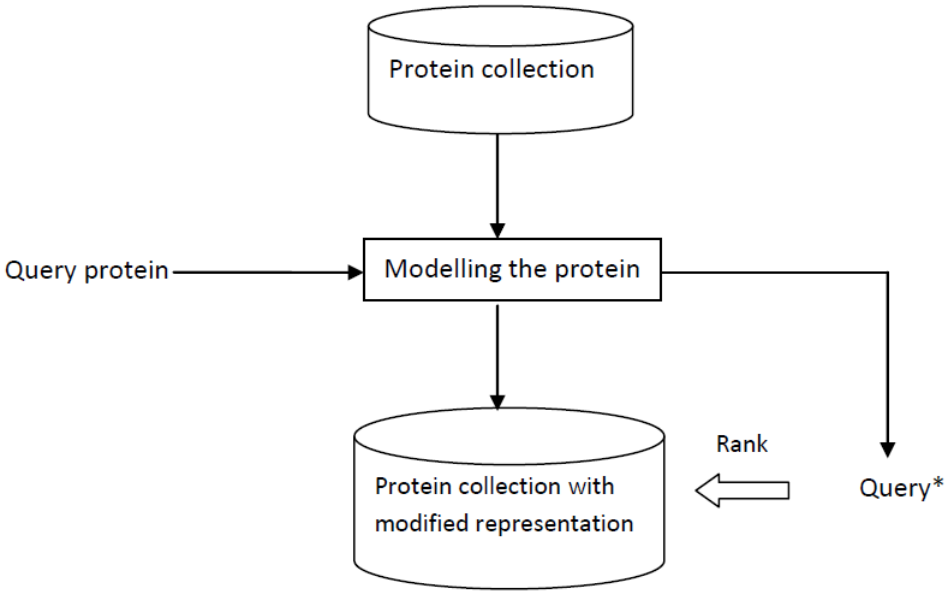

Figure 4.3: Typical Retrieval Model

As mentioned earlier, the retrieval might have different objectives for different applications. For example, retrieving proteins that are similar, whether they are close homologs or remote homologs. Text based IR researchers have shown that retrieval based on combination of multiple query representations, multiple representations of text documents, or multiple IR techniques provide significantly improved results compared to single representation based technique, especially when there are multiple retrieval requirements across users. These techniques are referred to as multi-view based IR in literature [1]. Schema of multi-view IR is given in Figure 4.4. The intuition behind doing this is: retrieval information about an author, publication or book would require exact keyword match, but querying based on topics, for example "sports news", must allow more than just keyword match.

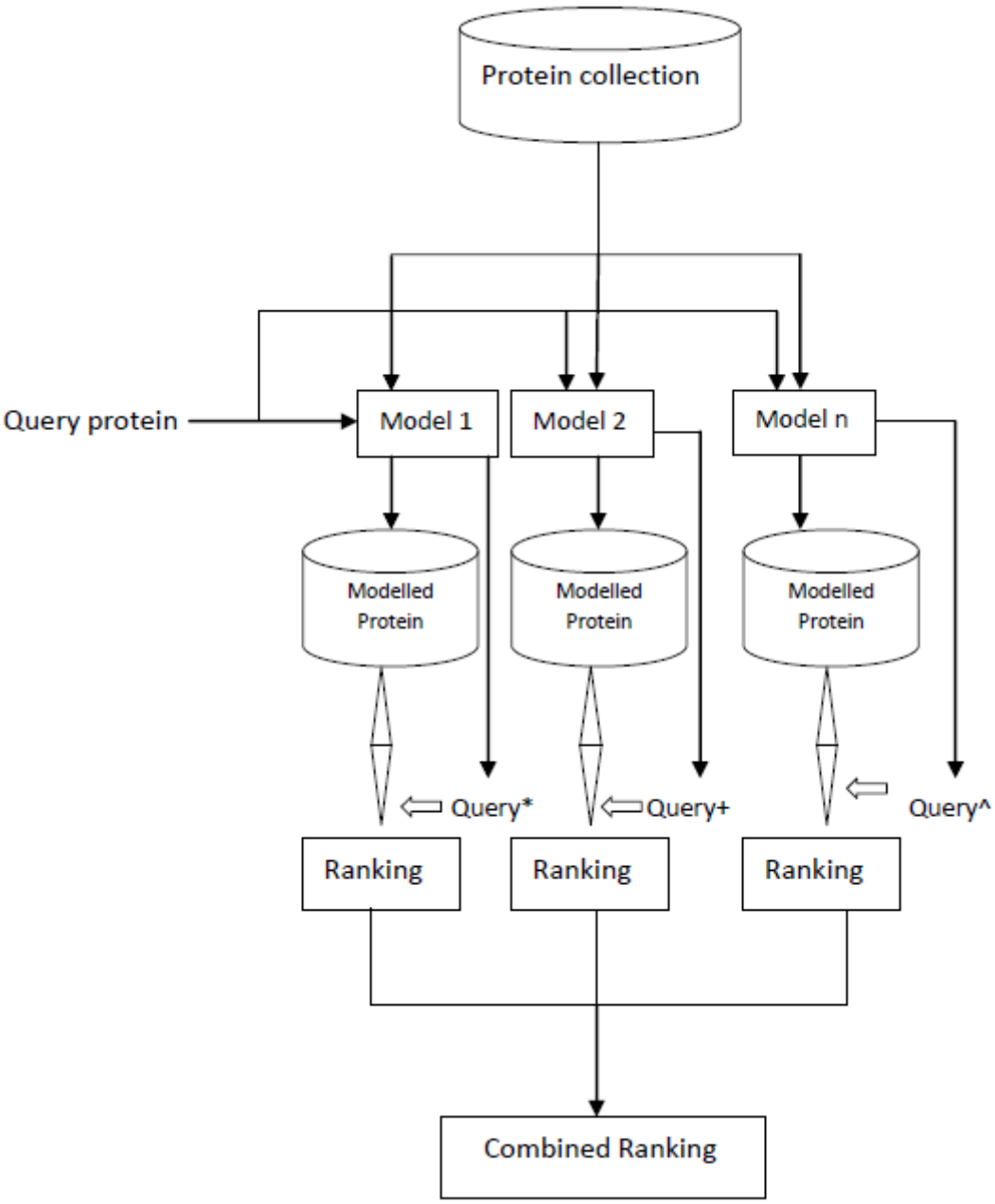

Figure 4.4: Multi-view based Retrieval Model

Motivated by the success of multi-view based text IR works, we propose a multi-view based retrieval system for protein structure collection. Protein structure similarity can be captured by not only matching fragments in the protein structure, but similar fragments (not just identity) must also be considered to help protein structure comparison. This is achieved by modeling the protein structure using LDA, which maps the fragments to a topic space using their co-occurrence information. Protein structure comparison at topic space performs a soft matching by considering similar fragments too.

The proposed model combines the plain vector (boolean or frequency based) representation of fragments in protein structure and topic space representation using LDA. Query protein and proteins in the collection are transformed into a naive vector space model and LDA representation. The retrieval techniques for both the modeling methods are different. Let us assume a simple boolean representation and a cosine similarity metric for the naive vector space model. Cosine similarity between two protein structures represented using boolean vectors A, B is given below

$$FragSimilarity(A, B) = cos(\theta) = \frac{A.B}{||A||||B||}$$

We refer to the similarity based on naive vector representation as *FragSimilarity*. LDA based representation uses the asymmetric Kullback Leibler(KL) divergence measure to rank proteins. Asymmetric KL divergence between two proteins represented by the topic distribution vector P and Q is given below

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}$$

The ranking based on these two techniques are combined using a weighted combination of similarity values. KL divergence captures the distance and not similarity value as in the case of cosine similarity. The range of values for cosine similarity$(0, 1)$ and KL divergence $(-\infty, 0)$ are different. KL divergence values are

normalized using min-max normalization to get normalized KL divergence measure D_{KL}^{norm}, and converted into similarity value by performing $1 - D_{KL}^{norm}$. Finally, the values are combined as follows

$$Similarity = \lambda_1 * FragSimilarity + \lambda_2 * (1 - D_{KL}^{norm})$$

λ_1 and λ_2 denote the relative weight for the retrieval schemes based on vector representation and LDA respectively. The model can also be extended to more representation schemes, where $0 \leq \lambda_i \leq 1$, and $\sum_i \lambda_i = 1$. Comparison of various vector representations and similarity metrics are given in the experimental results section.

4.3 Experimental Results

The experiments are performed on FragBag dataset [25] containing 2,930 sequence non-redundant structures selected from CATH version 2.4. The dataset was constructed by using a best-of-six structural aligner (using SSAP [68], STRUCTAL, DALI [32], LSQMAN [19], CE [23], and SSM) [25; 51]. The structural neighbors of each protein are determined using threshold on structural alignment score (SAS) obtained from the alignments. The following three SAS thresholds, 2 Å, 3.5 Å and 5 Å, are used to obtain close to remote homologs of the query protein. Each protein in the dataset is represented as a probability distribution over the latent topics discovered by LDA and is used as a query for evaluating performance of our method in ranking its structural neighbors. For constructing topic models, we use 7 out of 24 fragment libraries proposed by Kolodny et al. [51] based on their performance as reported in earlier work [25]. These seven libraries are as follows: 100(5), 300(6), 250(7), 600(9), 600(10), 400(11) and 400(12). Here each library is represented with the number of constituent fragments and their size. For example, 400(11) represents a library containing 400 fragments of size 11. The ranking performance is measured using area under the curve (AUC) of receiver operating characteristics

(ROC) curve. The overall AUC value is obtained by averaging individual AUC values across 2930 query proteins. The AUC takes values between 0 and 1 and its value closer to 1 indicates higher ranking performance.

A number of similarity measures exist for comparing protein structures in topic space. We compare the performance of the following distance measures: Cosine similarity (CO), Euclidean distance (EU) and KL divergence (KL) and select the one with the highest performance. The topics were discovered from proteins represented using fragments from 400(11) library, which is chosen due to its top performance in FragBag experiments [25]. Table 4.1 contains the results for three different SAS thresholds: 2 Å, 3 Å and 5 Å. It can be seen that KL and CO are more appropriate for SAS threshold of 2 Å and 5 Å, while KL performs slightly better than CO for SAS threshold of 3.5 Å. We use KL as a preferred distance measure for further analysis, since it outperforms the other two measures in most of the choices of the number of topics (Table 4.1).

	Topics								
Dist	10	100	150	200	250	300	400	500	SAS
KL	0.85	0.89	0.9	0.9	0.9	0.9	0.9	0.9	
EU	0.84	0.87	0.88	0.88	0.87	0.87	0.87	0.87	**2**
CO	0.85	0.89	0.89	0.89	0.9	0.9	0.9	0.9	
KL	0.71	0.77	0.77	0.78	0.77	0.78	0.78	0.77	
EU	0.69	0.71	0.73	0.73	0.73	0.73	0.72	0.72	**3.5**
CO	0.70	0.73	0.76	0.76	0.76	0.77	0.77	0.77	
KL	0.68	0.69	0.69	0.69	0.68	0.68	0.68	0.67	
EU	0.67	0.67	0.67	0.66	0.66	0.65	0.65	0.65	**5**
CO	0.68	0.69	0.69	0.69	0.69	0.69	0.69	0.68	

Table 4.1: Comparison of three different distance measures: Cosine similarity (CO), Euclidean distance (EU) and KL divergence (KL) based on average area under the curve (AUC) obtained by ranking structurally similar proteins, which are represented in topic space using 400(11) library.

In order to select the ideal number of topics, we represented proteins with various number of topics obtained for each of the seven fragment libraries and ranking performance of our method is obtained in terms of average AUC value for each of the SAS thresholds (Table 4.2). The analysis reveals that the representation using 200-250 topics has the best ranking performance. We use the best performing number of topics for each library in the further analyses.

	Topics								
Library	10	100	150	200	250	300	400	500	SAS
100(5)	0.83	0.85	0.86	0.88	0.88	0.87	0.86	0.84	
300(6)	0.84	0.85	0.86	0.88	0.88	0.88	0.87	0.85	
250(7)	0.85	0.87	0.88	0.89	0.9	0.89	0.89	0.89	
600(9)	0.85	0.89	0.9	0.9	0.9	0.9	0.89	0.88	**2**
600(10)	0.85	0.88	0.9	0.9	0.9	0.9	0.9	0.88	
400(11)	0.85	0.89	0.9	0.9	0.9	0.9	0.9	0.9	
400(12)	0.84	0.89	0.9	0.9	0.9	0.89	0.89	0.87	
100(5)	0.70	0.73	0.73	0.74	0.73	0.72	0.72	0.74	
300(6)	0.71	0.74	0.75	0.76	0.76	0.76	0.75	0.75	
250(7)	0.71	0.75	0.75	0.76	0.76	0.76	0.76	0.75	
600(9)	0.72	0.77	0.77	0.78	0.78	0.78	0.77	0.77	
600(10)	0.72	0.77	0.77	0.78	0.78	0.77	0.77	0.77	**3.5**
400(11)	0.71	0.77	0.77	0.78	0.77	0.78	0.77	0.77	
400(12)	0.71	0.77	0.77	0.77	0.76	0.76	0.76	0.76	
100(5)	0.67	0.68	0.68	0.68	0.67	0.67	0.67	0.67	
300(6)	0.66	0.67	0.68	0.68	0.68	0.68	0.68	0.67	
250(7)	0.66	0.69	0.69	0.68	0.68	0.68	0.67	0.67	
600(9)	0.68	0.69	0.7	0.69	0.68	0.68	0.68	0.67	
600(10)	0.67	0.69	0.69	0.69	0.68	0.68	0.68	0.66	**5**
400(11)	0.68	0.69	0.69	0.69	0.68	0.68	0.68	0.67	
400(12)	0.69	0.7	0.7	0.7	0.69	0.69	0.69	0.67	

Table 4.2: Selection of the best number of topics for representing proteins, using each of the seven libraries, based on their ranking performance indicated by the average AUC

As mentioned in Section 4.2.2, multi-view IR combines LDA and simple vector space model with weights λ_1 and λ_2. In this section, we identify the best simple vector space model from the following choices: (i) term frequency (TF), (ii) term frequency and inverse document frequency (TF-IDF), and (ii) boolean (Bool). We choose cosine similarity to compare vectors, since it has been shown to work well for such representations in literature. The AUC score for different λ values are given in Table 4.3. The values are computed on 400(11) library, which gives the best results across different number of LDA topics (from Table 4.2). The analysis of Table 4.3 reveals that the multi-view IR with either TF or TF-IDF as a simple vector space model performs better than FragBag [25]. Overall, combining TF and LDA gives the best results (Table 4.3). The experiments are repeated for other libraries using the best multi-view IR model (TF and LDA) and the results for SAS thresholds 2, 3.5 and 5 are given in Table 4.4, 4.5, 4.6 respectively.

It can be seen that for SAS thresholds of 2 and 3.5, best performance is achieved with a higher weight for LDA representation (λ_1). On the other hand, for SAS threshold of 5, best performance is achieved with a higher weight for simple vector space model (or at least equal to LDA representation). As mentioned earlier, SAS threshold of 2 tends to retrieve close homologs, 3.5 retrieves intermediate homologs and 5 denotes remote homologs. LDA based representation performs better in identifying close homologs (SAS threshold of 2 Å) than the remote ones (SAS threshold of 5 Å) for a given query protein. Since the fragment functionality overlap is less as we move up the parent tree for a protein structure, exact match using naive vector space model performs better than LDA representation to identify remote homologs.

	SAS=2			SAS=3.5			SAS=5		
λ_1	**I**	**II**	**III**	**I**	**II**	**III**	**I**	**II**	**III**
0	0.89	0.87	0.8	0.77	0.72	0.64	0.75	0.73	0.68
0.1	0.89	0.87	0.81	0.78	0.73	0.65	0.75	0.73	0.69
0.2	0.9	0.88	0.81	0.78	0.74	0.66	0.75	0.73	0.69
0.3	0.9	0.88	0.82	0.79	0.75	0.67	0.75	0.74	0.7
0.4	0.91	0.89	0.83	0.79	0.76	0.69	0.77	0.74	0.7
0.5	0.91	0.9	0.85	0.8	0.77	0.7	0.75	0.74	0.71
0.6	0.91	0.9	0.86	0.8	0.78	0.72	0.75	0.73	0.71
0.7	0.91	0.9	0.88	0.8	0.78	0.75	0.75	0.73	0.71
0.8	0.91	0.91	0.89	0.8	0.79	0.77	0.74	0.72	0.71
0.9	0.91	0.9	0.9	0.8	0.78	0.77	0.72	0.7	0.7
1	0.9	0.9	0.9	0.78	0.78	0.78	0.68	0.68	0.68

Table 4.3: Comparing the average AUC for various multi-view IR methods: The multi-view models are obtained by combining LDA with weight λ_1 and one of the following vector space models (i) term frequency (TF) (**I**), (ii) term frequency inverse document frequency (TF-IDF) (**II**) and (iii) boolean (BOOL) (**III**) with weight λ_2. Since $\lambda_2 = 1 - \lambda_1$, we have not mentioned their values explicitly in the table.

λ_1	400(12)	600(10)	600(9)	250(7)	200(6)	100(5)
0	0.88	0.88	0.88	0.87	0.85	0.86
0.1	0.89	0.89	0.89	0.88	0.86	0.86
0.2	0.89	0.89	0.89	0.88	0.86	0.86
0.3	0.9	0.9	0.9	0.88	0.86	0.86
0.4	0.9	0.9	0.9	0.89	0.87	0.86
0.5	0.9	0.91	0.91	0.89	0.88	0.87
0.6	0.91	0.91	0.91	0.89	0.88	0.87
0.7	0.91	0.91	0.91	0.9	0.89	0.87
0.8	0.91	0.91	0.91	0.9	0.89	0.87
0.9	0.9	0.91	0.91	0.9	0.89	0.87
1	0.89	0.9	0.9	0.89	0.88	0.87

Table 4.4: Comparison of models built on different libraries for SAS threshold of 2 Å: Here each library is denoted as $X(Y)$, where X is the number of fragments in the library, each of length Y. The ranking performance of a given multi-view IR model for a given library is given in terms of AUC. The multi-view model contains LDA model with weight λ_1 and TF vector space model with weight $1 - \lambda_1$

λ_1	400(12)	600(10)	600(9)	250(7)	200(6)	100(5)
0	0.76	0.76	0.76	0.74	0.69	0.72
0.1	0.77	0.77	0.77	0.75	0.7	0.72
0.2	0.78	0.77	0.77	0.75	0.71	0.72
0.3	0.78	0.78	0.78	0.76	0.72	0.73
0.4	0.79	0.78	0.79	0.76	0.73	0.73
0.5	0.79	0.79	0.79	0.76	0.74	0.73
0.6	0.79	0.8	0.8	0.77	0.75	0.74
0.7	0.8	0.8	0.8	0.77	0.75	0.74
0.8	0.8	0.8	0.8	0.78	0.76	0.74
0.9	0.79	0.79	0.8	0.77	0.76	0.75
1	0.77	0.77	0.78	0.76	0.76	0.74

Table 4.5: Comparison of models built on different libraries for SAS threshold of 3.5 Å: Here each library is denoted as $X(Y)$, where X is the number of fragments in the library, each of length Y. The ranking performance of a given multi-view IR model for a given library is given in terms of AUC. The multi-view model contains LDA model with weight λ_1 and TF vector space model with weight $1 - \lambda_1$

λ_1	400(12)	600(10)	600(9)	250(7)	200(6)	100(5)
0	0.76	0.75	0.75	0.75	0.71	0.73
0.1	0.76	0.75	0.75	0.75	0.72	0.73
0.2	0.76	0.75	0.75	0.75	0.72	0.73
0.3	0.76	0.76	0.76	0.76	0.72	0.73
0.4	0.76	0.76	0.76	0.76	0.73	0.73
0.5	0.76	0.76	0.76	0.76	0.73	0.73
0.6	0.76	0.76	0.76	0.76	0.72	0.73
0.7	0.76	0.75	0.75	0.75	0.72	0.72
0.8	0.75	0.74	0.74	0.74	0.71	0.72
0.9	0.73	0.73	0.73	0.73	0.7	0.71
1	0.69	0.69	0.69	0.69	0.68	0.68

Table 4.6: Comparison of models built on different libraries for SAS threshold of 5 Å: Here each library is denoted as $X(Y)$, where X is the number of fragments in the library, each of length Y. The ranking performance of a given multi-view IR model for a given library is given in terms of AUC. The multi-view model contains LDA model with weight λ_1 and TF vector space model with weight $1 - \lambda_1$

Motivated by the fact that the best results are spanning different libraries, a multi-view model which combines rankings based on representations using different libraries is proposed. For a given query protein structure, similarity produced by a model, say using library 400 (11), and weights $\lambda_1 = 0.6$, $\lambda_2 = 0.4$ is treated as an independent hypothesis. Output of each model (combination of libraries and λ_1, λ_2 values) is treated as a hypothesis. The best k hypotheses are empirically chosen and are combined using *bucket of models* strategy. For example, let $sim_X(q, s_d)$ and $sim_Y(q, s_d)$ be the similarity scores between a query protein q and a protein s_d in the database as provided by the models X and Y respectively. The similarity between q and s_d is given by $sim(q, s_d) = \max(sim_X(q, s_d), sim_Y(q, s_d))$. We refer to this as a *Combined Model*. We evaluated it by combining 3 best models across SAS thresholds. The best models are 600(9) with weights (0.8, 0.2) for SAS=2; 400(11) with weights (0.7, 0.3) for SAS=3.5; 400(11) with weights (0.4, 0.6) for SAS=5. Results for the *Combined Model* is given in Table 4.9.

In order to show the effectiveness of LDA based representation over Bag of Words (BoW) representation[25], we compare their performance on classification and clustering tasks. It can be seen that LDA representation performs better than the BoW for both the tasks, in terms of time taken and standard measures for the tasks. Table 4.7 has the comparison results for classification at C level classes in CATH hierarchy (4 classes). Since the dataset chosen is sparse at other levels of CATH hierarchy (has less than 10 members for most classes at A, T, H levels), we perform classification only at C level. Radial Basis Function network (RBF) and Naive Bayesian(NB) classifiers are used for the comparison. Results are compared in terms of root mean squared error (RMSE), ROC, and accuracy. The values are obtained by averaging results across 10 fold cross validation. Table 4.8 contains comparative results in terms of SSE (sum of squared error) for K Means algorithm using both BoW and LDA representations.

	BoW	LDA	BoW	LDA	BoW	LDA	BoW	LDA
	RMSE		ROC		Accuracy		Time (sec)	
RBF	0.25	0.23	0.93	0.95	83.9	85.7	6.84	2.5
NB	0.33	0.31	0.9	0.922	78.6	80.6	0.58	0.19

Table 4.7: Performance of BoW and LDA representations while classifying proteins at class (C) level of CATH classification.

	BoW	LDA
K	SSE	
4	8556.336	3371.61
10	8531.03	3417.21
20	8154.35	3348.29
50	7872.79	3093.44
100	7455.93	2880.1

Table 4.8: Performance of BoW and LDA for protein structure clustering task

The performance of LDA representation and retrieval based on asymmetric KL and multi-view retrieval using TF and LDA (multi-view model I) are compared against naive vector space model with cosine similarity on the chosen seven libraries. For multi-view based retrieval, the best weight combination of (λ_1 and λ_2) for each library is chosen for the plot. The results are shown in Figure 4.5, 4.6, 4.7 for SAS threshold of 2 Å, 3.5 Å and 5 Å respectively. Table 4.9 gives overall ranking of structural and filter methods, which includes the relative positioning of proposed techniques. Multi-view model I combines TF and LDA, Multi-view model II combines TF-IDF and LDA, Multi-view model III combines Boolean vector space model and LDA. It is clear that our method outperforms all the filter-and-match methods. We performed a paired t-test and paired sign test with AUC values of each query obtained using proposed models and baseline state-of-the-art filter-and-match method (FragBag). Based on the statistical test, our results are significantly better than the state-of-the-art at 1% significance level.

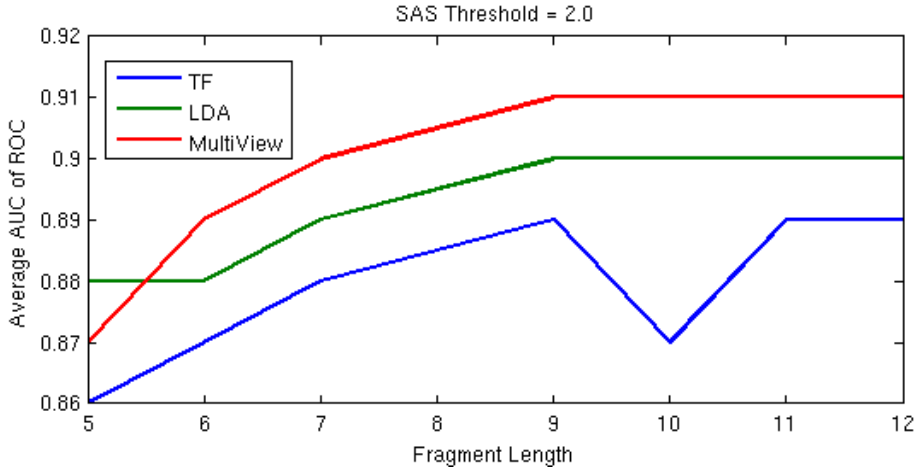

Figure 4.5: Comparison of the average AUC at SAS threshold of 2.0 Å, across libraries, obtained using TF, LDA and multi-view model using the best weights from Table 4.4

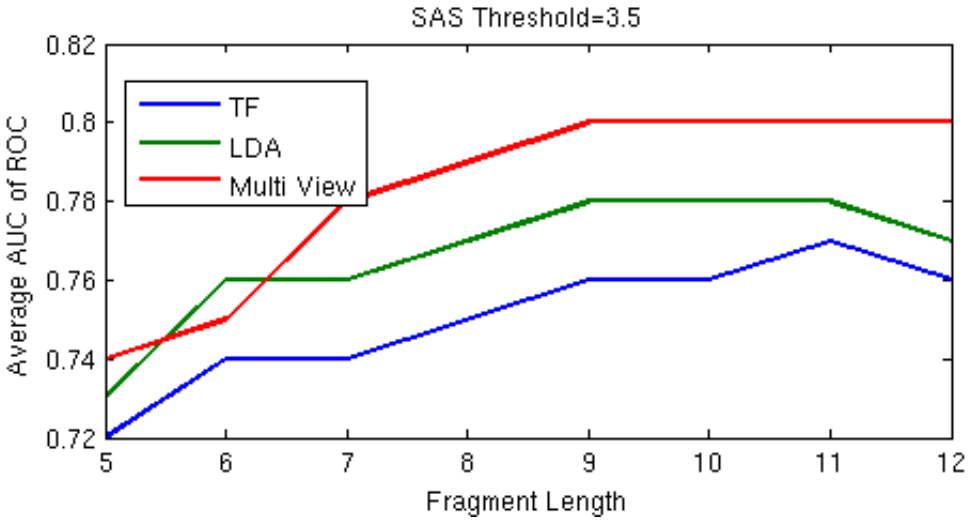

Figure 4.6: Comparison of the average AUC at SAS threshold of 3.5 Å, across libraries, obtained using TF, LDA and multi-view model using the best weights from Table 4.5

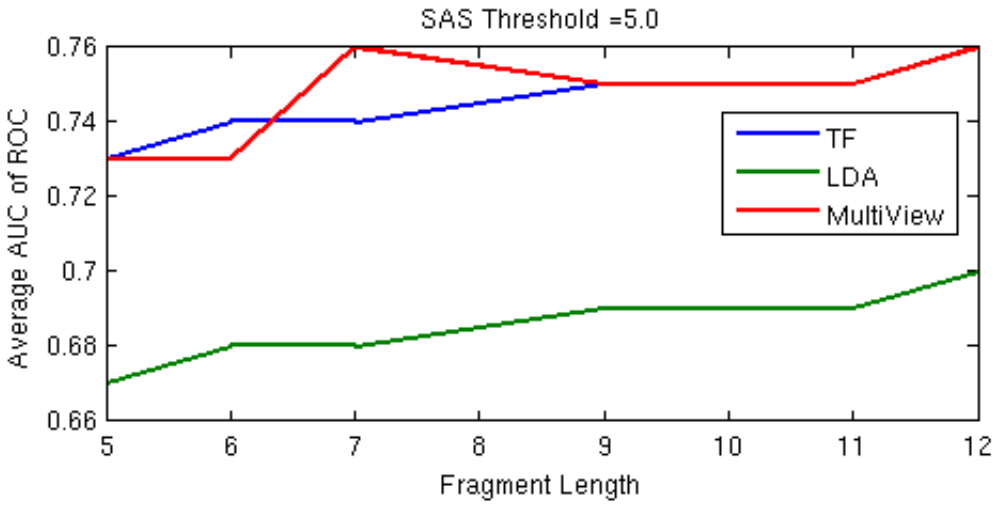

Figure 4.7: Comparison of the average AUC at SAS threshold of 5.0 Å, across libraries, obtained using TF, LDA and multi-view model using the best weights from Table 4.6

Methods	SAS=2	SAS=3.5	SAS=5	Average	Rank	Speed
SSM using SAS score	0.94	0.9	0.89	0.91	1	13
Structural using SAS score	0.9	0.81	0.84	0.85	2	39
Combined Model	0.92	0.82	0.75	**0.83**	3	Fast
Structural using native score	0.87	0.77	0.83	0.823	4	39
Multi-view model I (400,11)	0.91	0.8	0.76	**0.823**	4	Fast
CE using native score	0.9	0.79	0.74	0.81	6	54
Multi-view model II (400,11)	0.9	0.78	0.73	**0.803**	7	Fast
FragBag Cos distance (400,11)	0.89	0.77	0.75	0.803	7	Fast
Multi-view model III (400,11)	0.89	0.77	0.7	**0.787**	9	Fast
CE using SAS score	0.84	0.72	0.75	0.77	10	54
FragBag histogram intersection (600,11)	0.87	0.73	0.7	0.767	11	Fast
SGM	0.86	0.71	0.68	0.75	12	Fast
FragBag Euclidean distance (40,6)	0.86	0.71	0.64	0.737	13	Fast
Zotenko et al. (18)	0.78	0.64	0.66	0.693	14	Fast
Sequence matching by BLAST e-value	0.76	0.57	0.5	0.61	15	Fast
PRIDE	0.72	0.54	0.51	0.59	16	Fast

Table 4.9: AUCs of ROC Curves Using Best-of-Six Gold Standard: The proposed approaches are shown in bold. The speed is given as average CPU minutes per query. If the processing time (after preprocessing of protein structure) for a query is less than 0.1s, then it is mentioned as *fast*.

4.4 Conclusion

The contribution of this work includes a novel representation for protein structures and a multi-view based similar protein structures retrieval framework. We demonstrated that our method outperforms most of the existing filter-and-match methods. Our results are very competitive even with the state-of-the-art structure comparison methods operating at the level of complete three dimensional representation. Moreover, our method is much faster than these methods. Kolodny and co-workers first proposed the use of IR techniques in protein structure comparison [25]. In this work, we have shown significant improvements by adapting more powerful models from statistical NLP literature to this task. We have also taken advantage of multiple representations of the protein structure through the proposed multi-view based retrieval framework. This work has firmly established that such fragment based models can be competitive with the structural methods. It has also opened the doors for deeper analysis, using techniques from statistical NLP, of the role that fragments play in determining the overall structure.

CHAPTER 5

Conclusion and Discussions

In this chapter we summarize our contributions and point out the potential extensions for future research.

5.1 Conclusion

We have proposed multi-view approaches for linked data classification and similar protein structure retrieval tasks. First, we proposed a multi-view collective learning framework for sentiment analysis (MGSA). We performed sentiment analysis in a semi-supervised environment using both content and link information. Unlike other approaches which use collective classification for sentiment analysis, we do not assume to have fully labeled data[7; 40; 58]. Since we use content and link information separately, we call this as a multi-view based approach to perform sentiment analysis. The empirical analysis was performed on automobile reviews, and we found that MGSA outperformed the baseline classifiers. To overcome some of the key limitations of the framework proposed for sentiment analysis, we further proposed a two step multi-view collective learning framework (MVCL). MVCL involves cautious procedures for bootstrapping labels and learning classifiers on content and link views. MVCL performs better than MGSA and other single view approaches on Cora, Citeseer and WebKB linked datasets. Second, we proposed a multi-view retrieval framework for protein structures. We found that our method outperforms most of the existing filter-and-match methods. Our results are competitive even with the state-of-the-art structure comparison methods operating at the level of complete three dimensional representation. Moreover,

our method is much faster than these methods. We found that utilizing multiple redundant representations is more effective than using only one or combining the multiple representations and using it as one representation. The views might not be independent, but still redundancy among the representations and compatibility between the views help to achieve better performance than single view approaches[31]. The key contributions of this work are:

- Using link as a view in a multi-view learning setup to solve collective classification. We proposed cautious algorithms for bootstrapping and learning classifiers on the stronger content and the weaker link views.
- Exploiting multiple representations to solve the similar protein structures retrieval task.

5.2 Future Directions

Possible directions of future research include:

- We evaluated the multi-view learning framework on standard linked datasets. A thorough empirical study of the framework can be carried out on a variety of text domain datasets which possess more than two views. For example, an academic network data, in which the authors are connected to their co-authors and publications. Similarly each publication has its own citation network and is also connected to its authors.
- A theoretical analysis for using link as a view in multi-view learning framework can be performed[54].
- In the bootstrapping step, a better link view based label propagation technique that does not assume homophily can be used.
- We exploited multiple representations of protein structures for the retrieval task. It could be interesting to work on the multi-view based *learning* framework for the similar protein structures retrieval task (Learning to Rank framework for multi-view IR).
- We modelled protein structures using LDA. It might be useful to adapt more powerful NLP techniques to model the same. For example, proteins have a hierarchy of classes, so it might be effective to use hierarchical topic models to capture this rich information.

- We performed an overlapping fixed length segmentation of proteins into fragments. It would be effective to use multi-task topic models that can perform both segmentation of proteins into fragments and model the bag of fragments that can be used as a representation for protein structures[22].

Publications

1. Shivashankar, S., Ravindran, B., and Srinivasa Raghavan, N. R. "Text mining of Internet content: The bridge connecting product research with customers in the digital era." *In Product Research: The Art and Science Behind Successful Product Launches, Srinivasa Raghavan, N.R. and Cafeo, J. A. (Eds.), Nov. 2009. Springer*, pp. 231-242. (Invited Book Chapter)

2. Shivashankar, S., and Ravindran, B. "Multi-grain Sentiment Analysis using Collective Classification." *In the Proceedings of the Nineteenth European Conference on Artificial Intelligence (ECAI) 2010*, pp. 823-828. IOS Press.

3. Shivashankar, S., Srivathsan, S., Ravindran, B., and Tendulkar, A. V. "Multi-view Methods for Protein Structure Comparison using Latent Dirichlet Allocation". *In Bioinformatics 2011 27(13): pp.i61-i68. (Special issue with the proceedings of the 19th Annual International Conference on Intelligent Systems for Molecular Biology (ISMB/ECCB) 2011).*

4. Shivashankar, S., and Ravindran, B. "Multi-view Learning based Collective Classification : Leveraging Link view" *Under Preparation*.

REFERENCES

[1] Allison L. Powell and James C. French. 1998. The potential to improve retrieval effectiveness with multiple viewpoints. Technical report, University of Virginia, Charlottesville, VA, USA.

[2] Amr Ahmed and Eric P. Xing. 2010. Staying informed: supervised and semi-supervised multi-view topical analysis of ideological perspective. In *Proc of EMNLP*, 1140-1150.

[3] A. Sali and T. L. Blundell. 1990. Definition of general topological equivalence in protein structures. A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *J Mol Biol*, 212:403–428.

[4] B. Taskar, P. Abbeel, and D. Koller. 2002. Discriminative probabilistic models for relational data. In *Proc of UAI*, pp 485–492.

[5] Bing Liu. 2006. Web Data Mining. *Springer*.

[6] Blum, A., and Mitchell, T. 1998. Combining labeled and unlabeled data with co-training. In *Proc. of COLT*, pp. 92-100.

[7] Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts In *Proc. of ACL*, pp 271–278

[8] B. Krishnapuram, D. Williams, Y. Xue, A. Hartemink, L. Carin, and M. Figueiredo. 2004. On semi-supervised classication. In *Proc of NIPS*.

[9] B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques In *of Proc. of EMNLP*.

[10] Brian Gallagher, Hanghang Tong, Tina Eliassi-Rad, and Christos Faloutsos. 2008. Using ghost edges for classification in sparsely labeled networks. In *Proc of SIGKDD*, 256-264.

[11] B. W. Matthews, S. J. Remington, M. G. Grütter, and W. F. Anderson. 1981. Relation between hen egg white lysozyme and bacteriophage T4 lysozyme: Evolutionary implications. *J Mol Biol*, 147:545–558.

[12] Chi-Hua Tung, Jhang-Wei Huang, and Jinn-Moon Yang. 2007. Kappa-alpha plot derived structural alphabet and BLOSUM-like substitution matrix for rapid search of protein structure database. *Genome Biol*, 8:R31–R31.

[13] C. M. Christoudias, K. Saenko, L.-P. Morency, and T. Darrell. 2006. Co-adaptation of audio-visual speech and gesture classiers. In *Proc. of ICMI*.

[14] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

[15] David R. Hardoon and Kristiaan Pelcksman. 2010. Pair-Wise Cluster Analysis Technical report, arXiv:1009.3601v1

[16] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schlkopf. 2003. Learning with local and global consistency. In *Proc. of NIPS*.

[17] Elena Zotenko, Dianne P. O'Leary, and Teresa M. Przytycka. 2006. Secondary structure spatial conformation footprint: a novel method for fast protein structure comparison and classification. *BMC Struct Biol*, 6:12–12.

[18] G. Bakir, T. Hofmann, B. Schölkopf, A. Smola, B. Taskar, and S. V. N. Vishwanathan. 2007. Predicting Structured Data. In *Proc of MIT Press*, Cambridge, Massachusetts.

[19] G. J. Kleywegt. 1996. Use of non-crystallographic symmetry in protein structure refinement. *Acta Crystallogr., Sect. D: Biol. Crystallogr*, 52:842857.

[20] Geman, Stuart and Geman, Donald. 1984. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 721–741.

[21] H.-J. Oh, S. H. Myaeng, and M.-H. Lee. 2000. A practical hypertext catergorization method using links and incrementally available class information. In *Proc. of SIGIR*, pp 264–271

[22] Hemant Misra, Francois Yvon, Joemon M. Jose, and Olivier Cappe. 2009. Text segmentation via topic modeling: an analytical study. In *Proc. of CIKM*, pp 1553-1556.

[23] I. N. Shindyalov and P. E. Bourne. 1998. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng*, 11:739–747.

[24] Iddo Friedberg, Tim Harder, Rachel Kolodny, Einat Sitbon, Zhanwen Li, and Adam Godzik. 2007. Using an alignment of fragment strings for comparing protein structures. *Bioinformatics*, 23:e219–e224.

[25] Inbal Budowski-Tal, Yuval Nov, and Rachel Kolodny. 2010. FragBag, an accurate representation of protein structure, retrieves structural neighbors from the entire PDB quickly and accurately. *Proc Natl Acad Sci U S A*, 107:3481–3486.

[26] In-Geol Choi, Jaimyoung Kwon, and Sung-Hou Kim. 2004. Local feature frequency profile: a method to measure structural similarity in proteins. *Proc Natl Acad Sci U S A*, 101:3797–3802.

[27] J. Neville and D. Jensen. 2000. Iterative Classification in Relational Data. In *Proc. of the Workshop on Statistical Relational Learning, 17th National Conference on Artificial Intelligence*, pp. 4249.

[28] J. Neville and D. Jensen. 2007. Relational dependency networks. *Journal of Machine Learning Research*, 8:653692.

[54] Jennifer Neville and David Jensen. 2008. A bias/variance decomposition for models using collective inference. *Machine Learning*, 73, pp 87-106.

[30] Jain, Anil K., Robert P. W. Duin, and Jianchang Mao 2000. Statistical Pattern Recognition: A Review. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[31] Kamal Nigam and Rayid Ghani. 2003. Understanding the Behavior of Co-training. In *Proc. of KDD-2000 Workshop on Text Mining.*

[32] L. Holm and C. Sander. 1996. Mapping the protein universe. *Science*, 273:595–603.

[33] L. McDowell, K. M. Gupta, and D. W. Aha. 2007. Cautious inference in collective classification. In *Proc. of AAAI*, pp 596601.

[34] M. Balcan, A. Blum, and K. Yang. 2004. Co-training and expansion: Towards bridging theory and practice. In *Proc. of NIPS*.

[35] M. Collins and Y. Singer. 1999. Unsupervised models for named entity classication. In *Proc. of Joint SIGDAT Conference on EMNLP and Very Large Corpora*.

[36] M. E. Karpen, P. L. de Haseth, and K. E. Neet. 1989. Comparing short protein substructures by a method based on backbone torsion angles. *Proteins*, 6:155–167.

[37] M Zuker and R. L. Somorjai. 1989. The alignment of protein structures in three-dimensions. *Bulletine of Mathematical Biology*, 51:55–78.

[38] Manning, Christopher D. and Raghavan, Prabhakar and Schtze, Hinrich. 2008. Introduction to Information Retrieval. *Cambridge University Press*, isbn 0521865719, 9780521865715.

[39] Martin J. Wainwright and Michael I. Jordan 2008. Graphical Models, Exponential Families, and Variational Inference In *Proc of Foundations and Trends in Machine Learning*, v.1 n.1-2, p.1-305

[40] McDonald, Ryan , Hannan Kerry , Neylon Tyler , Wells Mike , Reynar Jeffrey C. 2007. Structured Models for Fine-to-Coarse Sentiment Analysis In *Proc. of ACL*, pp. 432-439

[41] McDowell, L.K., Gupta, K.M., and Aha, D.W. 2009. Cautious collective classification. In *Journal of Machine Learning Research.*

[42] Melville, P., Gryc, W., and Lawrence, R. D. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *of Proc. of SIGKDD*.

[43] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. In *Proc. of JMLR*, 7:2399–2434.

[44] Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building lexicon for Sentiment Analysis from Massive Collection of HTML Documents. In *of Proc. of EMNLP*.

[45] Peter D. Turney. 2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *of Proc. of ACL*.

[46] Peter D. Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. In *of Proc. of ACM Transactions on Information Systems*.

[47] Peter Rogen and Boris Fain. 2003. Automatic classification of protein structure by using Gauss integrals. *Proc Natl Acad Sci U S A*, 100:119–124.

[48] Prithviraj Sen, Galileo Mark Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. In *AI Magazine*, vol.29, no.3, pp 93–106.

[49] Qing Lu and Lise Getoor. 2003. Link-based Classification using Labeled and Unlabeled Data. In *Proc. of the ICML Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining.*

[50] Qiu, G., Liu, B., Bu, J., and Chen, C. 2009. Expanding domain sentiment lexicon through double propagation. In *of Proc. of IJCAI*.

[51] Rachel Kolodny, Patrice Koehl, and Michael Levitt. 2005. Comprehensive evaluation of protein structure alignment methods: scoring by geometric measures. *J Mol Biol*, 346:1173–1188.

[52] R. K. Ando and T. Zhang. 2007. Two-view feature generation model for semi-supervised learning. In *Proc of ICML*.

[53] S. Bickel and T. Scheffer. 2005. Estimation of mixture models using Co-EM. In *Proc. of ECML*.

[54] S. Dasgupta, M. Littman, and D. McAllester. 2001 PAC generalization bounds for co-training. In *Proc. of NIPS*.

[55] S. M. Kakade and D. P. Foster. 2007. Multi-view regression via canonical correlation analysis. In *of Proc. of COLT*.

[56] S. Yu, B. Krishnapuram, R. Rosales, H. Steck, and R. B. Rao. 2007. Bayesian co-training. In *of Proc. of NIPS*.

[57] Sofus A. Macskassy. 2007. Improving Learning in Networked Data by Combining Explicit and Mined Links. In *of AAAI*.

[58] Somasundaran, S., Namata, G., Wiebe, J., and Getoor, L. 2009. Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *of Proc. of EMNLP*.

[59] S. Chakrabarti, B. Dom, and P. Indyk. 1998. Enhanced hypertext categorization using hyperlinks. In *Proc. of SIGMOD*, Vol. 27, No. 2., pp. 307–318

[60] Thomas Gärtner. 2003. A survey of kernels for structured data. In *Proc of SIGKDD Explor. Newsl.* 5, 49-58.

[61] T. L. Griffiths and M. Steyvers. 2004 Finding scientific topics In *Proceedings of the National Academy of Sciences*, pp. 5228–5235.

[62] U. Brefeld and T. Scheffer. 2004. Co-EM Support Vector Learning. In *Proc of ICML*.

[63] Ulf Brefeld, Thomas Grtner, Tobias Scheffer and Stefan Wrobel. 2006. Efficient co-regularised least squares regression. In *Proc of ICML*.

[64] Virginia de Sa. 2005. Spectral clustering with two views. In *Proc. of ICML Workshop on Learning With Multiple Views*.

[65] Vikas Sindhwani and Partha Niyogi 2005. A co-regularized approach to semi-supervised learning with multiple views In *Proc. of the ICML Workshop on Learning with Multiple Views*

[66] V. Sindhwani and P. Melville. 2008. Document-word co-regularization for semi-supervised sentiment analysis. In *of Proc. of ICDM*

[67] V. Sindhwani and D. Rosenberg. 2008. An RKHS for Multi-view Learning and Manifold Co-Regularization. In *of Proc. of ICML*.

[68] W. R. Taylor and C. A. Orengo. 1989. Protein structure alignment. *J Mol Biol*, 208:1–22.

[69] William R Taylor, Alex C W May, Nigel P Brown, and Andras Aszodi. 2001. Protein structure: geometry, topology and classification. *Reports on Progress in Physics*, 64:517–590.

[70] X. Zhu. 2005. Semi-supervised learning literature survey. Technical Report 1530, Department of Computer Sciences, University of Wisconsin, Madison.

[71] X. Zhu, Z. Ghahramani, and J. D. Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. of ICML*, pp 912–919.

[72] Yasemin Altun, David McAllester, and Mikhail Belkin. 2006. Maximum margin semi-supervised learning for structured variables. In *Proc. of NIPS*, pp 33–40.