Edge Weight Regularization Over Multiple Graphs For Similarity Learning

Pradeep Muthukrishnan
Department of Electrical Engineering and Computer Science
University of Michigan,
Ann Arbor, USA
Email: mpradeep@umich.edu

Dragomir Radev, Qiaozhu Mei
School of Information, Dept of EECS
University of Michigan
Ann Arbor, USA
Email: radev@umich.edu, qmei@umich.edu

***Abstract*—The growth of the web has directly influenced the increase in the availability of relational data. One of the key problems in mining such data is computing the similarity between objects with heterogeneous feature types. For example, publications have many heterogeneous features like text, citations, authorship information, venue information, etc. In most approaches, similarity is estimated using each feature type in isolation and then combined in a linear fashion. However, this approach does not take advantage of the dependencies between the different feature spaces. In this paper, we propose a novel approach to combine the different sources of similarity using a regularization framework over edges in multiple graphs. We show that the objective function induced by the framework is convex. We also propose an efficient algorithm using coordinate descent [1] to solve the optimization problem. We extrinsically evaluate the performance of the proposed *unified* similarity measure on two different tasks, clustering and classification. The proposed similarity measure outperforms three baselines and a state-of-the-art classification algorithm on a variety of standard, large data sets.**

***Keywords*-Machine Learning; Similarity Learning; Heterogeneous Features; Classification; Clustering**

I. Introduction

Nowadays, relational data are universal and have a broad appeal in many different application domains.The recent upsurge in the amount of text available due to the widespread growth of the Internet has led to the need for large scale, efficient Natural Language Processing (NLP), Information Retrieval (IR) tools for text mining. At the heart of many of the NLP, IR algorithms is the need for a good similarity measure. For example, many of the algorithms for Document Clustering, Word Sense Disambiguation, Document classification, etc make use of similarity measures.

Similarity graphs are estimated from data and may not accurately reflect the semantic relationship between the objects. It is trivially clear that the more accurate the similarity graph reflects the true semantic relationship better the performance of tasks like classification, clustering and other data mining tasks.

In the past, there was a huge advantage of the text being homogeneous (pure text) and hence it was easier to compute similarity. For example, there exists many different similarity measures for text [18], [12], [13]. But currently, almost all data can be represented using much more than text. For example, publications have many heterogeneous features like text, citations, authorship information, publication venue information, etc. But there are very few similarity measures that use all the different relations *together*. In most cases, similarity is computed using just one feature type or similarity is computed using two feature types individually and then combined in a linear weighted fashion.

In the case of publications, text is the most commonly used feature for computing similarity. Recently, there has been a lot of interest in using all the different feature types to compute similarity. For example, [2] use both citations and text to compute similarity between publications . They compute textual similarity using cosine similarity. Two papers are considered to be similar if one paper cites the other. They combine the similarity between the two paper due to citation (CS) and text (TS) in a linear fashion as follows

$$S(x, y) = \alpha TS(x, y) + (1 - \alpha)CS(x, y) \qquad (1)$$

where, $\alpha \leq 1$ and $S(x, y)$ represents the combined similarity between publications x and y.

This formulation clearly does not take into account any dependency between the different feature spaces (in this case, text and citation space).

Also, even in the case of the same feature space, higher order dependencies are not taken into account. For example, let publication X contain three words $\{w_1, w_2\}$ and publication Y contain $\{w_2, w_3\}$ then this indicates that words w_1 and w_3 are similar to each other and should affect the similarity between two other publications containing them.

Nevertheless, [20] takes these dependencies into account and represents each document in the classical vector space model and they use a Naive Bayes classifier to classify text documents. The strength of the classifier lies in the augmented representation of the documents. The new representation includes the words present in the document as well as words which are present in higher order dependencies. For example, in the classical vector space model, publication X's vector representation would include only w_1 and w_2, but not w_3. Whereas, in the new representation all the words are weighted by the number of higher order paths which contain the word. They demonstrate that including

higher order dependencies improves classification accuracy and most of the information required for classification is contained in second order dependencies.

[21] gives a mathematical proof for the dependence of Latent Semantic Indexing (LSI)[22] on higher order dependencies. That is, if two documents X and Y have an initial similarity of zero and if the similarity becomes non-zero after using LSI, then it implies that there exists a higher order dependency between the two documents. Higher order co-occurrences have also been used earlier in Word Sense Disambiguation [23] and stemming [24]

Although, there has been some work on using higher order dependencies it is not easily extended for multiple feature spaces, especially for heterogeneous feature types like citations, text, etc. In the case of using citations and text, it is not immediately clear how to model higher order dependencies when using multiple feature types. Also, another difficulty in extracting higher order dependencies is in including more than one feature space.

One possible approach to use all feature types is to make a big graph using all the features in the same graph. For example, in the case of publications, the set of nodes can include the papers and keywords. The edges could represent citations between papers and the relation between papers and keywords. This approach is equivalent to merging multiple classifiers (each using a different feature type/view) into a single classifier using all the feature types which is not recommended. There could be data points where each of the classifiers are "authoritative" on but by merging them this confidence is diluted. Also, [27] proves that a classifier has low generalization error if it agrees on unlabeled data with a second classifier based on different "view" of the data. This justifies the search for multiple feature types to be represented separately.

Our work is also quite similar to co-training [26] in the sense of using multiple views (feature types). Also, [25] proposes a greedy algorithm to search for classifiers which have provable advantages over the classifiers found using the co-training algorithm. The greedy algorithm searches for complex classifiers which are essentially a list of atomic rules **H**, each assigning a label *l* based on the presence of a single feature *h*. However, even this approach does not take care of dependencies across feature types and higher-order dependencies between the same feature type. Our algorithm seeks to estimate improved similarity measures by integrating all available sources of similarity.

II. Problem Motivation

We seek to motivate the need for an *unified* similarity measure using an example. In this example, we consider only two different feature types, but it naturally extends to more than two feature types. Consider the problem of classification of publications by research area. Let there be three publications, P_1, P_2, P_3 in the area of Machine Translation. Let the publications contain two keywords, $k_1 =$" Statistical Machine Translation" and $k_2 =$"Bilingual Corpora". Specifically, publications P_1 contains k_1, P_2 and P_3 contain k_2. Although the two keywords are enough for a human to clearly see that the two papers are on Machine Translation, no textual similarity measure (without using external knowledge sources) will assign a non-zero similarity value between any pair of publications. In general, this is a very commonly occurring problem in text mining.

However, there are other sources of similarity that can be useful in this case. It is likely that two papers in the same area are similar due to the structure of citation graph. For example, it is possible that P_1 and P_2 are both cited by a lot of papers thereby inducing some similarity between P_1 and P_2. Assume that P_1 and P_2 are similar due to citation information. This information can be used to induce a small amount of similarity between the keywords contained in the two papers. Thus, keywords k_1 and k_2 have a non-zero similarity value. Once again, this information can be used to induce a small amount of similarity between P_1 and P_3. Therefore, two features which co-occur a lot are similar and similarly, two objects which have a lot of similar features are similar themselves. Thus, we can learn to estimate similarity in one feature space using similarity estimates from other feature space. The major contributions of this paper are given below,

- A simple representation model for representing objects with heterogeneous feature types for efficiently estimating similarity.
- A regularization framework for unifying different sources of similarity
- Two completely unsupervised algorithms in the proposed framework to efficiently compute similarity for objects with many heterogeneous feature types.

In the next two sections, we formalize and exploit this observation to improve similarity in one feature space using other feature spaces in a principled way.

III. Problem Formulation

We assume that the data to be analyzed consists of many heterogeneous feature types. The setup is very general: The objects are represented by many heterogeneous feature types and an initial coarse similarity measure between different feature types as well as objects is estimated from the data. The estimated similarity values are represented using multiple layers of graphs, where each graph corresponds to a particular feature type.

We now formally define the related concepts,

Definition 1. Objects and Feature Types: we have a set of N objects,
$O = \{o_1, o_2, \ldots, o_N\}$. Each object is represented using m different feature types, $F = \{F_1, F_2, \ldots, F_m\}$. Each feature type, F_i consists of a set of features,

$F_i = \{f_{1i}, f_{2i}, \ldots, f_{n_i i}\}$.

Definition 2. Object Graph: The object graph, $G_0 = \{V_0, E_0\}$, consists of the set of objects as nodes and the edge weights represent the initial similarity between the objects.

Definition 3. Feature Graphs: For each feature type F_i, we create a feature graph, $G_i = (V_i, E_i)$. The graph consists of features F_i as nodes, i.e, $V_i = F_i$. The edge weights represent the similarity between features. In the case of feature types like citations/links, the set of nodes in the graph is V_0 and similarity can be estimated using the citation/link graph structure using node similarity measures [3], [14].

The object graph and the feature graphs can be viewed as different layers of graphs. These layers are connected using the concept of layer connectivity as defined below,

Definition 4. Layer Connectivity: The two graphs, G_i and G_j, $i \neq j$ are connected as follows. There exists an edge between $f_i j$ and $f_k l$ if they co-occur in some object's feature vector representation. Let Z refer to the layer connectivity function, i.e, $Z_{f_{ij}, f_{kl}} = 1$ if $f_i j$ and $f_k l$ co-occur.

Thus we have a set of heterogeneous graphs each with an initial similarity measure. We seek to improve the similarity values in each graph layer using information from all other layers. Specifically, we incorporate higher order dependencies in the same feature type as well as dependencies across feature types to improve the initial similarity estimates.

we define two features, f_{ik} and f_{jk} of the feature type F_k to be similar if

- Two objects which contain both the features are similar, i.e, o_x contains f_{ik}, o_y contains f_{jk} and o_x and o_y are similar.
- Two features of a different type which co-occur with these features are similar, i.e, f_{xl} co-occurs with f_{ik}, f_{yl} co-occurs with f_{jk} and f_{xl} and f_{yl} are similar.

and similarly, object similarity can be defined in terms of feature similarity. Therefore, the task is to obtain improved similarity estimates in all feature spaces. We evaluate the estimated similarity measure extrinsically, i.e, using performance measures for external tasks like classification and clustering which use the improved similarity measure.

IV. Regularization Framework

Our approach for incorporating information from different heterogeneous feature types is inspired by the standard regularization framework for semi-supervised classification using label propagation [15], [16]. In this framework, there exists a single graph $G = (V, E, w)$. The set of nodes is represented as $V = \{x_1, x_2, \ldots, x_n\}$. The edge weights, w_{ij} represent similarity between the two nodes i and j. Also, there exists a set of labeled nodes L whose label is given by $y(x)$. The problem is to classify all other nodes using a discriminant function $f : X \rightarrow \mathbb{R}$. To compute f they define an objective function as follows,

$$\Omega(f) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f(\mathbf{x_i}) - f(\mathbf{x_j}))^2 + \mu \sum_{x \in L} (f(\mathbf{x}) - y(\mathbf{x}))^2 \tag{2}$$

Essentially, the first term tries to regularize the node labeling over the network such that similar nodes get similar labels while the second term tries to minimize the difference between the true labels and the predicted labels. The first term can be written in quadratic form as shown below,

$$\Omega(f) = \frac{1}{2}\mathbf{f}^T \mathcal{L} \mathbf{f} + \mu \sum_{x \in L} (f(\mathbf{x}) - y(\mathbf{x}))^2 \tag{3}$$

where $\mathbf{f} = (f(\mathbf{x_1}, f(\mathbf{x_2}, \ldots, f(\mathbf{x_n})$ and $\mathcal{L}$ is the combinatorial graph laplacian matrix, defined as $= D - W$ where D is the diagonal matrix with $d_{ii} = \sum_{j=1}^{n} w_{ij}$

One reasonable approach to combine the different sources of similarity for performing classification is to linearly combine the graph laplacians. For this purpose, we can create m different graphs $G_1, G_2, \ldots, G_m$, where each graph, G_i consists of the set of objects as nodes and edge weights represent similarity computed using only feature type F_i. Let the laplacian corresponding to graph G_i be represented as $\mathcal{L}_i$. Then we can perform regularization on all the graphs simultaneously as

$$\Omega(f) = \frac{1}{2}\sum_{i=1}^{m} \alpha_i \mathbf{f}^T \mathcal{L}_i \mathbf{f} + \mu \sum_{x \in L} (f(\mathbf{x}) - y(\mathbf{x}))^2 \tag{4}$$

where $\sum_i \alpha_i = 1$. This is equivalent to linearly combining the similarity estimates due to different feature types as in 1. Therefore, even this formulation does not use all the sources of similarity efficiently. There are a few problems with the above framework in the context of our problem setting

- The above framework does not take into account the edges introduced by layer connectivity and hence does not use all the graphs *together*.
- The regularization is defined over nodes in the graph, whereas for improving similarity estimates we want to define a regularizer over edge weights.

In this paper, we define a novel regularization framework over edge weights of multiple graphs for improved similarity estimation. Let $\mathcal{W}$ refer to the improved weighting function to be computed over the edges in all the graphs, i.e, $w_{f_{ik}, f_{jk}}$ gives the similarity between features f_{ik} and f_{jk} while w^*_{ij} refers to the initial similarity value. Also, let $V = V_o \cup F_1 \cup F_2 \ldots F_m$ refer to the set of vertices in all the graphs. initial similarity. Then, the objective function is defined as,

$$\Omega(w) = \alpha_0 \sum_{u,v \in V} (w_{u,v} - w^*_{u,v})^2 + \sum_{i,j=0}^{m} \alpha_{ij} \mathcal{J}(V_i, V_j) \quad (5)$$

where

$$\mathcal{J}(V_i, V_j) = \sum_{u_1, v_1 \in V_i} \sum_{u_2, v_2 \in V_j} Z_{u_1,u_2} Z_{v_1,v_2} (w_{u_1,v_1} - w_{u_2,v_2})^2 \quad (6)$$

and $\alpha_0 + \sum_{i,j=0}^{m} \alpha_{ij} = 1$

Then the task is to minimize the objective function. The objective function consists of two parts. The first part ensures the optimized similarity values remain close to the original similarity values while the second term seeks to minimize the dissimilarity between all the graphs. The second term directly models our intuition of features co-occurring with other features are similar. The parameter $0 < \alpha_0 < 1$ is used to control the tradeoff between the two terms. Note that if $\alpha_0 = 1$ then the optimal solution is the initial solution itself.

The significance of the second term is explained using a simple example. Consider two graphs, G_1 and G_2. Let G_1 be the graph containing publications as nodes and edge weights representing similarity due to citation graph. Let G_2 be the graph corresponding to keywords and edge weights represent similarity between keywords. There is an edge from a node u_1 in G_1 to a node v_1 in G_2 if the publication corresponding to u_1 contains the keyword corresponding to v_1. According to this example, minimizing the objective function essentially means updating similarity values between two keywords in proportion to similarity values between the papers they are contained in and vice versa. Therefore, even though keywords like *Machine Translation* and *Word Alignment* are not similar according to textual similarity measures, they are assigned a similarity value if two papers which contain them are similar because of citation similarity.

A. Convex Optimization - Direct Solution

Clearly, the objective function is a convex function in M variables where M is the total number of edges, i.e, $M = \sum_{i=1}^{m} |E_i|$. We can also add the constraint $\sum_v W_{u,v} = 1$ for normalization. Then the constrained convex optimization problem can be expressed in the form shown below,

$$minimize \frac{1}{2} x^T P x + Qx + r \quad (7)$$
$$subject\ to\ Ax = b$$

where x is a $M \times 1$ vector representing the edge weights along each dimension of the vector. The Karush-Kuhn-Tucker (KKT) conditions for the above problem are [17],

$$Ax^* = b,\ Px^* + q + A^T v^* = 0, \quad (8)$$

which can be written as,

$$\begin{bmatrix} P & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ v^* \end{bmatrix} = \begin{bmatrix} -q \\ b \end{bmatrix} \quad (9)$$

In the above equation, v^* refers to the dual solution while x^* refers to the actual solution. Hence, we can directly solve the optimization function to obtain the similarity values by solving the above system of linear equations. Unfortunately, solving the above system of equations requires inverting a $2M * 2M$ matrix which has a computationally complexity of $O(M^3)$. This is prohibitively expensive in terms of computational complexity for even moderate size data sets. Hence we need to find alternate optimization methods to minimize the objective function.

B. An Efficient Algorithm

Although the direct optimization is prohibitively expensive, we can solve it using approximate methods such as coordinate descent[1]. In this method, we assume all other dimensions are fixed except one and make a descent along the chosen dimension iteratively. In the context of our problem, we assume all edge weights are fixed except one edge weight and solve the minimization of the objective function. For example, the partial derivative of the objective function with respect to the edge, $(u_j, v_k) \in V_i$ is shown below,

$$\frac{\partial \Omega(w)}{\partial w_{u_j,v_k}} = 2\alpha_0 (w_{u_j,v_k} - w^*_{u_j,v_k}) + 2 \sum_{l=0,1,\ldots,i-1,i+1,\ldots,m} \alpha_{il} \sum_{u_2,v_2 \in V_l} Z_{u_j,u_2} Z_{v_k,v_2} (w_{u_j,v_k} - w_{u_2,v_2}) \quad (10)$$

To perform the descent along the edge $(u_j, v_k) \in V_i$, we set the above partial derivative to zero which gives us the following expression,

$$w_{u_j,v_k} = \frac{1}{C} \alpha_0 w^*_{u_j,v_k} + \sum_{l=0,1,\ldots,i-1,i+1,\ldots,m} \alpha_{il} \sum_{u_2,v_2 \in V_l} Z_{u_j,u_2} Z_{v_k,v_2} w_{u_2,v_2} \quad (11)$$

where C is defined as

$$C = \alpha_0 + \sum_{l=0,1,\ldots,i-1,i+1,\ldots,m} \alpha_{il} \sum_{u_2,v_2 \in V_l} Z_{u_j,u_2} Z_{v_k,v_2} \quad (12)$$

It can be seen that if the original similarity values are bounded in the range $[0, 1]$, then they will always remain bounded with C playing the role of a normalization constant. Note that C is a constant for a given pair of vertices and

hence, C can be incorporated in w^* and w to obtain the following equation

$$w_{u_j,v_k} = \alpha_0 w^*_{u_j,v_k} + \sum_{l=0,1,\ldots,i-1,i+1,\ldots,m} \alpha_{il} \sum_{u_2,v_2 \in V_l} Z_{u_j,u_2} Z_{v_k,v_2} w_{u_2,v_2} \quad (13)$$

If we represent the initial edge weights of graph G_i by matrix W_i^* and $Z_i j$ represent the layer connectivity matrices for graphs, G_i and G_j, then the algorithm can be written in terms of iterative matrix equations as follows,

$$W_i^t = \alpha_0 W_i^* + \sum_{l=0,1,\ldots,i-1,i+1,\ldots,m} Z_{il} W_l^{t-1} {Z_{il}}^T \quad (14)$$

where W_i^t refers to the matrix W_i at the end of the t^{th} iteration and ${Z_{il}}^T$ is the transpose of Z_{il}. Now, we can iteratively apply the above equation to minimize the objective function. See appendix for a proof of convergence for the algorithm.

For example, in the context of publications, let T represent the similarity between keywords and P represent the similarity between publications due to the citation graph. Then we can combine the two different sources of similarity using the following two iterative equations

$$\begin{aligned} T_t &= \alpha * T^* + (1-\alpha) * (Z.P^{t-1}.Z^T) \\ P_t &= \alpha * P^* + (1-\alpha) * (Z^T.kT_{t-1}.Z) \end{aligned} \quad (15)$$

C. Layered Random Walk

The above algorithm has a nice intuitive interpretation in terms of random walks over different layers of the graph assming the initial similarity weights are transition probability values and Z_{ij} matrices are normalized so that each row sums to 1. Then the similarity between two nodes, u and v in layer i, is computed as the sum of two parts. The first part is α_0 times the original edge weight. This is necessary so that the newly computed similarity estimates are not too far away from the initial estimates. The second part is a sum of $m-1$ different terms. The j^{th} term is weighted by $\alpha_i j$ and is the probability of starting a random walk from u and moving to layer j for a random walk of length 1 and then returning back to vertex v in layer i. Thus, the j^{th} term is the probability of a random walk of length 3 with both intermediate vertices of the walk belonging to layer j.

We can see that this algorithm exploits the dependencies across feature spaces to improve similarity estimates. Consider the example of publication classification mentioned in section II. When we estimate similarity between the keywords in the keyword feature layer, we perform a random walk over the citation feature layer. Therefore, the similarity between the keywords k_1 and k_2 will be incremented by a value directly proportional to the similarity between the publications due to citations. Also, note that second and higher order dependencies are also taken into account by this algorithm. That is, two papers may become similar because they contain two keywords which are connected by a path in the keyword feature layer, whose length is greater than 1. This is due to the iterative nature of the algorithm. For example, consider a set of semantically similar keywords $K = \{k_1, k_2, \ldots, k_n\}$. Assume that the initial keyword similarity estimated the similarity between these keywords is 0. Then during the first iteration, k_1 and k_2 could become similar because they are contained in publications P_1 and P_2 (similar due to citation similarity). Then k_2 could become similar to k_3 because of another pair of similar papers which contain the keywords. In this fashion, the similarity between all the keywords in the set are improved to a non-zero similarity value.

Also, the α_{ij} values can be used to control the user knowledge about the dependency between features. For example, it might be reasonable to assume that there is relatively low dependency between the venue information and authorship information. In this case, we can set the corresponding value of $alpha_{ij}$ to be low. This shows that we can model a large number of dependencies between different feature spaces using the rich representation using heterogeneous graphs which are interconnected.

V. Experiments

It is very hard to evaluate similarity measures in isolation. Thus, most of the algorithms to compute similarity scores are evaluated extrinsically, i.e, the similarity scores are used for an external task like clustering or classification and the performance in the external task is used as the performance measure for the similarity scores. This also helps demonstrate the different applications of the computed similarity measure. Thus, we perform a variety of different experiments on standard data sets to illustrate the improved performance of the proposed similarity measure.

Experiment Set I: We compare our similarity measure against other similarity measures in the context of classification. We also compare against a state of the art classification algorithm which uses different similarity measures due to different feature types without integrating them into one single similarity measure.

Specifically, we compare our algorithm against three other similarity baselines in the context of classification which are listed below.

- *Content Similarity*: Similarity is computed using just the feature vector representation using just the text. We use cosine similarity after preprocessing each document into a tf.idf vector for the AAN data set. For all other data sets, we use the cosine similarity on the binary feature vector representation that is available.
- *Link Similarity*: Similarity is computed using only the links (citations, in the case of publications). To compute

ACL-ID	Paper Title	Research Topic
W05-0812	Improved HMM Alignment Models for Languages With Scarce Resources	Machine Translation
P07-1111	A Re-Examination of Machine Learning Approaches for Sentence-Level MT Evaluation	Machine Translation
P03-1054	Accurate Unlexicalized Parsing	Dependency Parsing
P07-1050	K-Best Spanning Tree Parsing	Dependency Parsing
P88-1020	Planning Coherent Multi-Sentential Text	Summarization

Table I
DETAILS OF A FEW SAMPLE PAPERS CLASSIFIED ACCORDING TO RESEARCH TOPIC

Data Set	#Nodes	#Edges	#Keywords	#Classes
AAN	380	3	572	4221
WebKB	877	1608	1703	5
Cora	2708	5429	1433	7

Table II
STATISTICS OF DIFFERENT DATA SETS

link similarity, we use the node similarity algorithm proposed by Harel et al [3] using a random walk of length 3 on the link graph.

- *Linear combination*: The content similarity and link similarity are combined in a linear fashion as shown in equation 1. We tried different values of α and report only the best accuracy that can be achieved using linear combination of similarity measures.

We use a semi-supervised graph classification algorithm [5] to perform the classification given the different similarity graphs mentioned above.

We also compare our algorithm against

SC-MV: We compare our algorithm against the spectral clustering algorithm for data with multiple views [9]. The algorithm tries to classify data when multiple views of the data are available. The multiple views are represented using multiple homogeneous graphs with a common vertex set, V. In each graph, the edge weights represent similarity between the nodes computed using a single feature type. For our experiments, we used the link similarity graph and the content similarity graph as described above as the two views of the same data.

Experiment Set II: We illustrate the improved performance of our similarity measure in the context of clustering. We compare our similarity measure against the three similarity baselines mentioned above. We use a spectral graph clustering algorithm proposed in [10] to perform the clustering.

For all experiments, we set $\alpha_0 = 0.4$. This is because the initial similarity estimates are very coarse and computed using naive methods like cosine similarity. Hence, there was a bigger weight on the similarity learned from other feature spaces. We performed our experiments on three different data sets. The three data sets are explained below.

- AAN Data: The ACL Anthology is a collection of papers from the Computational Linguistics journal as well as proceedings from ACL conferences and workshops and includes $15,160$ papers. To build the **A**CL **A**nthology **N**etwork (AAN), [4] manually performed some preprocessing tasks including parsing references and building the network metadata, the citation, and the author collaboration networks. The full AAN includes the raw text of all the papers in addition to full citation and collaboration networks.
We chose a subset of papers in 3 topics (Machine Translation, Dependency Parsing, Summarization) from the ACL anthology. These topics are three main research areas in Natural Language Processing (NLP). Specifically, we collected all papers which were cited by papers whose titles contain any of the following phrases, "'Dependency Parsing"', "'Machine Translation"', "'Summarization"'. From this list, we removed all the papers which contained any of the above phrases in their title because this would make the clustering task easy. The pruned list contains 1190 papers. We manually classified each paper into four classes (Dependency Parsing, Machine Translation, Summarization, Other) by considering the full text of the paper. The manually cleaned data set consists of 275 Machine Translation papers, 73 Dependency Parsing papers and 32 Summarization papers. Table I lists a few sample papers from each class.
WebKB[19]: The data set consists of a subset of the original WebKB data set. The corpus consists of 877 web pages collected from four different universities. Each web page is represented by a 0/1-valued word vector with 1703 unique words after stemming and removing stopwords. All words with document frequency less than 10 were removed.
Cora[19]: The Cora dataset consists of 2708 scientific publications classified into one of seven classes. The citation network consists of 5429 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1433 unique words.

TableII summarizes the statistics of the different data sets

Machine Translation	Dependency Parsing	Summarization
Machine Translation	Natural Language	Rhetorical Structure
Statistical Machine Translation	Wall Street Journal	Novel Approach
Machine Translation System	Penn Treebank	System for generating
Translation Quality	Dependency Structures	Summarization System
Word Alignment	Dependency Structure Analysis	Source Document
Bleu Score	Kyoto University Corpus	Newspaper Articles
Parallel Texts	Dependency Tree	Multidocument Summarization
Experimental Results	Shared Task	Headline Generation System
Parallel Corpora	Dependency Grammar	Important Sentences
Language Pairs	Parsing Accuracy	Proposed Method
Target Language	Training Data	Discourse Structure
IBM Model	Lexical items	Document Compressions

Table III
TOP FEW WORDS FROM EACH CLUSTER EXTRACTED FROM THE KEYWORD SIMILARITY GRAPH

used in experiments

For all the data sets, we constructed two graphs, the kewyord feature graph and the link similarity graph. The keyword feature layer graph, $G_f = (V_f, E_f, w_f)$ is a weighted graph where V_f is the set of all features. The edge weight between the features f_i and f_j represents the similarity between the features. The edge weights are initialized to the cosine similarity between their corresponding document vectors. The link similarity graph, $G_o = (V_o, E_o, w_o)$ is another weighted graph where V_o is the set of objects. The edge weight represents the similarity between the objects and is initialized to the similarity between the objects due to the link structure. The link similarity between two objects is computed using the similarity measure proposed by [3] on the object link graph.

We evaluate our algorithm in terms of classification accuracy for the classification task. Classification accuracy is the percentage of objects which are correctly classified.

For the clustering task, we use Normalized Mutual Information as the measure of clustering accuracy. Mutual Information is a symmetric measure which quantifies the statistical information between two distributions. Let $I(X,Y)$ denote the amount of mutual information between the distributions X and Y. Since I(X,Y) has a lower bound of zero and no upper bound, we normalize the quantity using the entropy of X and Y. Thus the normalized mutual information used is,

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} \tag{16}$$

We estimate NMI using the samples provided by the clusterings. Let n be the total number of objects (nodes) to be clustered. Let $k(A)$ be the number of clusters according to clustering A and let $k(B)$ be the number of clusters according to clustering B. Let n_h^A denote the number of objects in cluster C_h according to clustering A , and let n_l^B denote the number of objects in cluster C_l according to clustering B . Let $n_{h,l}$ denote the number of objects that are in cluster C_h according to algorithm A as well as in group C_l according to clustering B. Then, NMI is estimated according to 16

$$\widehat{NMI(A,B)} = \frac{\sum_{h=1}^{k(A)}\sum_{l=1}^{k(B)} n_{h,l} log(\frac{n.n_{h,l}}{n_h^A n_l^B})}{\sqrt{\sum_{h=1}^{k(A)} \frac{n_h^A}{n} \sum_{h=1}^{k(B)} \frac{n_h^B}{n}}} \tag{17}$$

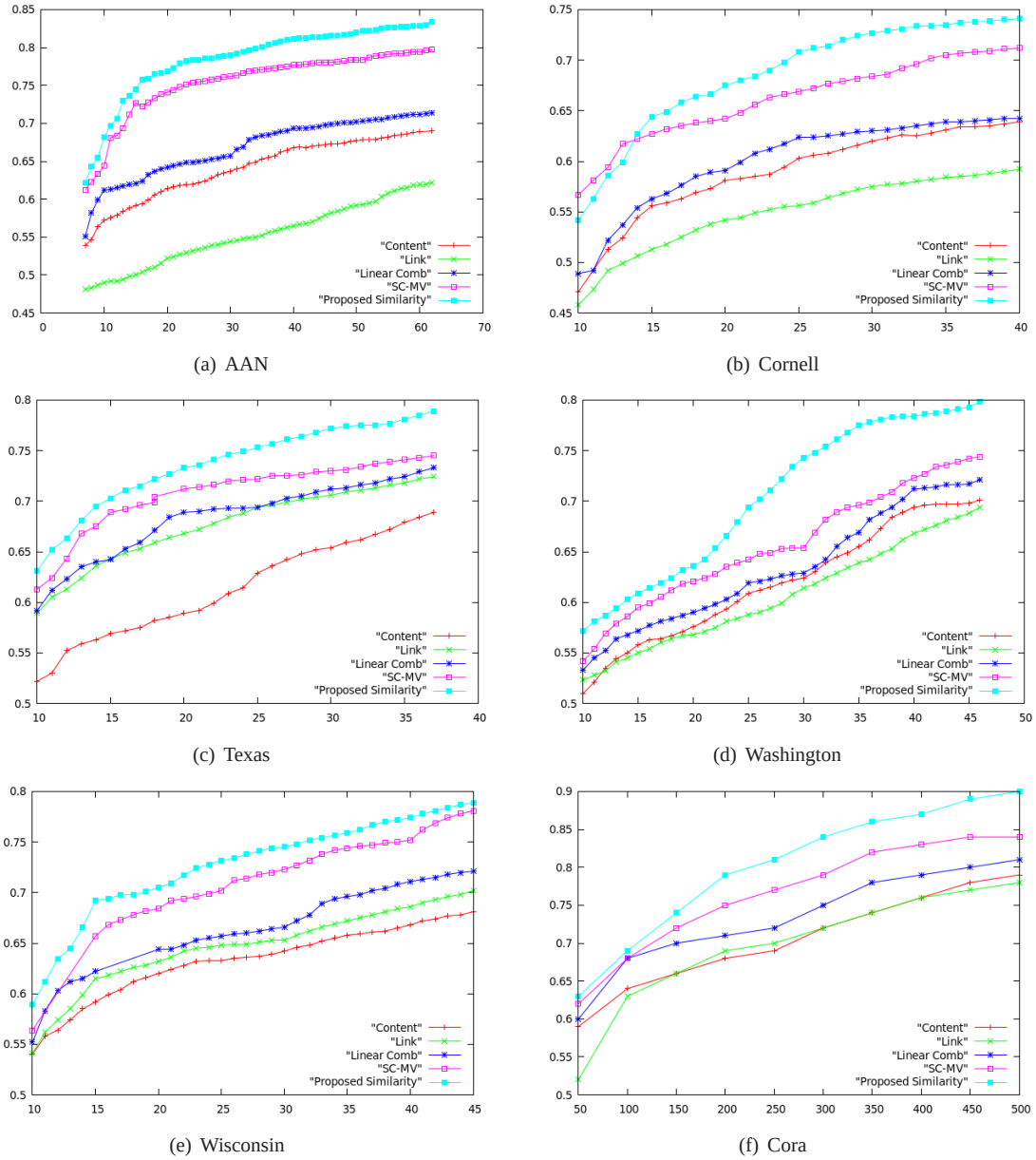
This evaluation measure has some nice properties such as $NMI(X,X) = 1$. Also $\widehat{NMI(X,Y)} = 1$ if the clusterings X and Y are the same except for a permutation of the cluster labels. Also, $NMI(X,Y) = 0$ if the clustering X is random with respect to Y or vice versa.

Therefore, in our setting, we compare the clusterings obtained by our algorithm X against the clustering Y, induced by the correct cluster labels using NMI. The higher the value the better the clustering obtained. Thus, we compare our algorithm against the two above mentioned baselines using NMI as the evaluation metric.

VI. RESULTS

Figure V shows the accuracy of the classification obtained using different similarity measures.

It can be seen that the proposed similarity measure outperforms all other baselines by a large margin. Note that although the spectral clustering algorithm on multiple graphs is given the same information as given to our algorithm, our algorithm achieves better performance. We attribute this to the rich representation of the data. In our algorithm, the data is represented as a set of heterogeneous graphs (layers) which are connected together. Thus, we were able to iteratively improve our similarity estimates using similarity estimates from other feature spaces. Whereas, in the case of the algorithm in [9] all the graphs are isolated homogeneous



(a) AAN (b) Cornell

(c) Texas (d) Washington

(e) Wisconsin (f) Cora

Figure 1. Classification Accuracy on the different data sets. The number of points labeled is plotted along the x-axis and the y-axis shows the classification accuracy on the unlabeled data.

Similarity Measure	AAN	Texas	Wisconsin	Washington	Cornell	Cora
Content Similarity (Cosine)	0.66	0.34	0.42	0.59	0.63	0.48
Link Similarity	0.45	0.49	0.39	0.52	0.56	0.52
Linear Combination	0.69	0.54	0.46	0.54	0.68	0.54
Unified Similarity	**0.78**	**0.69**	**0.54**	**0.66**	**0.72**	**0.64**

Table IV
NORMALIZED MUTUAL INFORMATION SCORES OF THE DIFFERENT SIMILARITY MEASURES ON THE DIFFERENT DATA SETS

graphs. Hence there is no information transfer across the different graphs.

Table IV shows the NMI scores obtained by the different similarity measures on the different data sets.

We also clustered the keyword feature layer of the AAN data set to show that the proposed algorithm improves the

similarity estimates in all the input graphs. Table III shows the most frequent keywords from each cluster. It can be seen that the keyword clusters are very cohesive and are indicative of the research area they belong to. This shows that similarity estimates in the keyword feature layer are also improved using similarity estimates from the citation similarity space.

An important property of this framework is that we enhance any coarse-grained initial similarity measure. For example, similarity between keywords can be estimated using more sophisticated methods which use external knowledge bases like WordNet, Wikipedia [6], [7]. Although, there exist lots of similarity measures for computing similarity between keywords and phrases, there are relatively few approaches to compute similarity between entire documents using pairwise similarities between keywords [8]. Note that using the pairwise similarities between individual keywords alone, we can estimate the similarity between entire documents using the proposed algorithm. We plan to do experiments in future comparing the proposed algorithm to other algorithms in this domain which already use a very sophisticated similarity measure in a single feature space (text).

VII. Conclusion

In this paper, we proposed a novel approach to compute similarity for objects with many heterogeneous features. We formalized the problem of similarity estimation as an optimization problem induced by a regularization framework over edges in multiple graphs. We show that there exists a direct solution to the convex optimization problem, albeit expensive in terms of computational complexity. Therefore, we proposed an alternate algorithm for the optimization using the coordinate descent approach. We also illustrated the improved performance of the proposed similarity measure in tasks like clustering and classification over baseline similarity measures and a state-of-the-art classification algorithm which uses the same information as given to our algorithm.

Acknowledgments

The authors would like to thank Vahed Qazvinian, Ahmed Hassan, Kumar Sricharan and Gowtham Bellala for helpful discussions.

References

[1] Z. Q. Luo and P. Tseng, "On the convergence of the coordinate descent method for convex differentiable minimization," *Journal of Optimization Theory and Applications*, vol. 72, no. 1, pp. 7–35, 1992.

[2] A. Hassan and D. R. Radev, "A mixture of networks approach to combining link based and content based networks," in *In Preparation*, 2010.

[3] D. Harel and Y. Koren, "On clustering using random walks," in *Foundations of Software Technology and Theoretical Computer Science 2245*. Springer-Verlag, 2001, pp. 18–41.

[4] D. R. Radev, P. Muthukrishnan, and V. Qazvinian., "The ACL Anthology Network corpus." 2009.

[5] X. Z. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *In Proceedings of the International Conference on Machine Learning*, 2003, pp. 912–919.

[6] M. I. Danushka Bollegala, Yutaka Matsuo, "Measuring semantic similarity between words using web search engines," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*. New York, NY, USA: ACM, 2007, pp. 757–766.

[7] M. Strube and S. P. Ponzetto, "Wikirelate! computing semantic relatedness using wikipedia," in *AAAI'06: proceedings of the 21st national conference on Artificial intelligence*. AAAI Press, 2006, pp. 1419–1424.

[8] R. Mihalcea, C. Corley, and C. Strapparava, "Corpus-based and knowledge-based measures of text semantic similarity," in *AAAI'06: Proceedings of the 21st national conference on Artificial intelligence*. AAAI Press, 2006, pp. 775–780.

[9] D. Zhou and C. J. C. Burges, "Spectral clustering and transductive learning with multiple views," in *ICML '07: Proceedings of the 24th international conference on Machine learning*. New York, NY, USA: ACM, 2007, pp. 1159–1166. [Online]. Available: http://dx.doi.org/10.1145/1273496.1273642

[10] I. S. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors a multilevel approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 11, pp. 1944–1957, November 2007. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2007.1115

[11] A. Strehl and J. Ghosh, "Cluster ensembles: a knowledge reuse framework for combining partitionings," in *Eighteenth national conference on Artificial intelligence*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 93–98. [Online]. Available: http://dx.doi.org/10.1016/S0167-8655(99)00104-X

[12] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," Tech. Rep. 8, 1966.

[13] C. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: a string kernel for svm protein classification." *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pp. 564–575, 2002. [Online]. Available: http://view.ncbi.nlm.nih.gov/pubmed/11928508

[14] G. Jeh and J. Widom, "Simrank: a measure of structural-context similarity," in *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM Press, pp. 538–543.

[15] Q. Mei, D. Cai, D. Zhang, and C. Zhai, "Topic modeling with network regularization," in *WWW '08: Proceeding of the 17th international conference on World Wide Web*. New York, NY, USA: ACM, 2008, pp. 101–110.

[16] A. B. Goldberg, X. Zhu, and S. Wright, "Dissimilarity in graph-based semi-supervised classification," *Journal of Machine Learning Research*, vol. 2, pp. 155–162, 2007.

[17] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0521833787

[18] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[19] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, pp. 93–106, 2008.

[20] M. C. Ganiz, N. I. Lytkin, and W. M. Pottenger, "Leveraging higher order dependencies between features for text classification," in *ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 375–390.

[21] A. Kontostathis and W. M. Pottenger, "A framework for understanding latent semantic indexing (lsi) performance," *Information Processing and Management*, vol. 42, no. 1, pp. 56–73, 2006.

[22] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, pp. 391–407, 1990.

[23] H. Schütze, "Automatic word sense discrimination," *Computational Linguistics*, vol. 24, no. 1, pp. 97–123, 1998.

[24] J. Xu and W. B. Croft, "Corpus-based stemming using cooccurrence of word variants," *ACM Transactions on Information Systems*, vol. 16, no. 1, pp. 61–81, 1998.

[25] S. Abney, "Bootstrapping," in *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2002, pp. 360–367.

[26] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *COLT' 98: Proceedings of the eleventh annual conference on Computational learning theory*. New York, NY, USA: ACM, 1998, pp. 92–100.

[27] S. Dasgupta, M. L. Littman, and D. Mcallester, "Pac generalization bounds for co-training," 2001. [Online]. Available: http://ttic.uchicago.edu/~dmcallester/cotrain01.ps

VIII. Appendix

A. Proof of Convergence

We can prove that equations 15 converge as the number of iterations tends to infinity.

Proof: In order to prove convergence, we need to show the following

$$W_t - W_{t-1} \xrightarrow{t\to\infty} 0 \tag{18}$$

$$W_t - W_{t-1} = (1-\alpha)Z(P_{t-1} - P_{t-2})Z^T \tag{19}$$

$$P_{t-1} = \alpha P_0 + (1-\alpha)Z^T W_{t-2} Z \tag{20}$$

$$P_{t-2} = \alpha P_0 + (1-\alpha)Z^T W_{t-3} Z \tag{21}$$

Thus,

$$P_{t-1} - P_{t-2} = (1-\alpha)\lambda Z^T (W_{t-2} - W_{t-3})Z$$

Substituting this in equation 4,

$$\begin{aligned} W_t - W_{t-1} \\ &= (1-\alpha)^2 ZZ^T (W_{t-2} - W_{t-3}) ZZ^T \\ &= (1-\alpha)^4 (ZZ^T)^2 (W_{t-4} - W_{t-5}) (ZZ^T)^2 \\ &\vdots \\ &= (1-\alpha)^t (ZZ^T)^{\frac{t}{2}} (W_1 - W_0) (ZZ^T)^{\frac{t}{2}} \end{aligned}$$

In the above equation, since $\alpha < 1$ we can claim that $W_t - W_{t-1} = 0$ as $t \to \infty$ if none of the matrices tend to ∞. $W_1 - W_0$ is a constant matrix and ZZ^T is a transition probability matrix, where $ZZ^T(i,j)$ is the probability of random walk of length starting at i and ending at j of length 2 . Hence $(ZZ^T)^{\frac{t}{2}}$ converges to the stationary probability distribution as $t \to \infty$. Hence,$W_t - W_{t-1} = 0$ as $t \to \infty$. □