Simultaneous Similarity Learning and Feature-Weight Learning for Document Clustering

Pradeep Muthukrishnan
Dept of CSE,
University of Michigan
mpradeep@umich.edu

Dragomir Radev
School of Information,
Dept of CSE,
University of Michigan
radev@umich.edu

Qiaozhu Mei
School of Information,
Dept of CSE,
University of Michigan
qmei@umich.edu

Abstract

A key problem in document classification and clustering is learning the similarity between documents. Traditional approaches include estimating similarity between feature vectors of documents where the vectors are computed using TF-IDF in the bag-of-words model. However, these approaches do not work well when either similar documents do not use the same vocabulary or the feature vectors are not estimated correctly.

In this paper, we represent documents and keywords using multiple layers of connected graphs. We pose the problem of *simultaneously* learning similarity between documents and keyword weights as an edge-weight regularization problem over the different layers of graphs. Unlike most feature weight learning algorithms, we propose an unsupervised algorithm in the proposed framework to simultaneously optimize similarity and the keyword weights. We extrinsically evaluate the performance of the proposed similarity measure on two different tasks, clustering and classification. The proposed similarity measure outperforms the similarity measure proposed by (Muthukrishnan et al., 2010), a state-of-the-art classification algorithm (Zhou and Burges, 2007) and three different baselines on a variety of standard, large data sets.

1 Introduction

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 Machine Learning (ML), Information Retrieval (IR) tools for text mining. At the heart of many of the ML, IR algorithms is the need for a good similarity measure between documents. For example, a better similarity measure almost always leads to better performance in tasks like document classification, clustering, etc.

Traditional approaches represent documents with many keywords using a simple feature vector description. Then, similarity between two documents is estimated using the dot product between their corresponding vectors. However, such similarity measures do not use all the keywords *together* and hence, suffer from problems due to sparsity. There are two major issues in computing similarity between documents

- Similar documents may not use the same vocabulary.
- Estimating feature weights or weight of a keyword to the document it is contained in.

For example, consider two publications, X and Y, in the field of Machine Learning. Let X be a paper on clustering while Y is on classification. Although the two publications use very different vocabulary, they are semantically similar. Keyword weights are mostly estimated using the frequency of the keyword in the document. For example, TF-IDF based scoring is the most commonly used approach to compute keyword weights to compute similarity between documents. However, suppose publications X and Y mention the keyword "'Machine Learning"' only once. Although, they are mentioned only once in the text of the document, for the purposes of computing semantic similarity between the docu-

Proceedings of the TextGraphs-6 Workshop, pages 42–50,
Portland, Oregon, USA, 19-24 June 2011.

ments, it would be beneficial to give it a high keyword weight.

A commonly used approach to estimate semantic similarity between documents is to use an external knowledge source like WordNet (Pedersen et al., 2004). However, these approaches are domain dependent and language dependent. If document similarity can not be estimated accurately using just the text, there have been approaches incorporating multiple sources of similarity like link similarity, authorship similarity between publications (Bach et al., 2004; Cortes et al., 2009). (Muthukrishnan et al., 2010) also uses multiple sources of similarity. In addition to improving similarity estimates between documents, it also improves similarity estimates between keywords. Co-clustering (Dhillon et al., 2003) based approaches are used to alleviate problems due to the sparsity and high-dimensionality of the data. In co-clustering, the keywords and the documents are *simultaneously* clustered by exploiting the duality between them. However, the approach relies solely on the keyword distributions to cluster the documents and vice-versa. It does not take into account the inherent similarity between the keywords (documents) when clustering the documents (keywords). Also, co-clustering takes as input the weight of all keywords to corresponding documents.

2 Motivation

First, we explain how similarity learning and feature weight learning can mutually benefit from each other using an example. For example, consider the following three publications in the field of Machine Translation, (Brown et al., 1990; Gale and Church, 1991; Marcu and Wong, 2002)

Clearly, all the papers belong to the field of *Machine Translation* but (Gale and Church, 1991) contains the phrase "'Machine Translation"' only once in the entire text. However, we can learn to attribute some similarity between (Brown et al., 1990) and the second publication using the text in (Marcu and Wong, 2002). The keywords "'Bilingual Corpora"' and "'Machine Translation"' co-occur in the text in (Marcu and Wong, 2002) which makes the keywords themselves similar. Now we can attribute some similarity between the (Brown et al., 1990) and (Marcu and Wong, 2002) publication since they contain similar keywords. This shows how similarity learning can benefit from important keywords.

Now, assume that "'Machine Translation"' is an *important* keyword (high keyword weight) for the first and third publication while "'Bilingual Corpora"' is an important keyword for the second publication. We explained how to infer similarity between the first and second publication using the third publication as a bridge. Using the newly learned similarity measure, we can infer that "'Bilingual Corpora"' is an *important* keyword for the second publication since a similar keyword ("'Machine Translation"') is an important keyword for similar publications.

Let documents D_i and D_j contain keywords K_{ik} and K_{jl}. Then intuitively, the similarity between two documents should be jointly proportional to

- The similarity between keywords K_{ik} and K_{jl}
- The weights of K_{ik} to D_i and K_{jl} to D_j.

Similarly the weight of a keyword K_{ik} to document D_i should be jointly proportional to

- The similarity between documents D_i and D_j.
- The similarity between keyphrases K_{ik} and K_{jl} and weight of K_{jl} to D_j.

The major contributions of the paper are given below,

- A rich representation model for representing documents with associated keywords for efficiently estimating document similarity..
- A regularization framework for *joint* feature weight (keyword weight) learning and similarity learning.
- An unsupervised algorithm in the proposed framework to efficiently learn similarity between documents and the weights of keywords for each document in a set of documents.

In the next two sections, we formalize and exploit this observation to jointly optimize similarity between documents and weight of keywords to documents in a principled way.

3 Problem Formulation

We assume that a set of keywords have been extracted for the set of documents to be analyzed. The setup is very general: Documents are represented by the set of candidate keywords. In addition to that, we have crude initial similarities estimated between documents and also between keywords and the weights of keywords to documents. The similarities and keyword weights are represented using two layers of graphs. We formally define the necessary concepts,

Definition 1: Documents and corresponding keywords
We have a set of N documents $D = \{d_1, d_2, \ldots, d_N\}$. Each document, d_i has a set of m_i keywords $K_i = \{k_{i1}, k_{i2}, \ldots, k_{im_i}\}$

Definition 2: Document Similarity Graph
The document similarity graph, $G_1 = (V_1, E_1)$, consists of the set of documents as nodes and the edge weights represent the initial similarity between the documents.

Definition 3: Keyword Similarity Graph
The keyword similarity graph, $G_2 = (V_2, E_2)$, consists of the set of keywords as nodes and the edge weights represent the initial similarity between the keywords.

The document similarity graph and the keyword similarity graph can be considered as two layers of graphs which are connected by the function defined below

Definition 4: keyword Weights (KW)
There exists an edge between d_i and k_{ij} for $1 \leq j \leq m_i$. Let Z represent the keyword weighting function, i.e, $Z_{d_i,k_{ij}}$ represents the weight of keyword k_{ij} t document d_i.

4 Regularization Framework

$$\begin{aligned}\Omega(w, Z) \quad &= \alpha_0 * ISC(w, w*) + \alpha_1 * IKC(Z, Z*) \\ &+\alpha_2 * KS(w, Z) + \alpha_3 * SK(Z, w) \quad (1)\end{aligned}$$

where $\alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 = 1$.

ISC refers to **I**nitial **S**imilarity **C**riterion and *IKC* refers to **I**nitial **K**eyword weight **C**riterion. They are defined as follows

$$ISC(w, w*) = \sum_{u,v \in G_1} (w_{u,v} - w^*_{u,v})^2 \quad (2)$$

$$IKC(Z, Z*) = \sum_{u \in G_1, v \in G_2} (Z_{u,v} - Z^*_{u,v})^2 \quad (3)$$

KS refers to **K**eyword based **S**imilarity and *SK* refers to **S**imilarity induced **K**eyword weight. They are defined as follows

$$KS(w, Z) = \sum_{u_1,v_1 \in G_1} \sum_{u_2,v_2 \in G_2} Z_{u_1,u_2} Z_{v_1,v_2} (w_{u_1,v_1} - w_{u_2,v_2})^2 \quad (4)$$

and

$$SK(w, Z) = \sum_{u_1,v_1 \in G_1} \sum_{u_2,v_2 \in G_2} w_{u_1,v_1} w_{u_2,v_2} (Z_{u_1,u_2} - Z_{v_1,v_2})^2 \quad (5)$$

Then the task is to minimize the objective function. The objective function consists of four parts. The first and second parts are initial similarity criterion and initial keyword criterion. They ensure that the optimized edge weights are close to the initial edge weights. The weights α_0 and α_1 ensure that the optimized weights are close to the initial weights, in other words, they represent the confidence level in initial weights.

The significance of the third and the fourth parts of the objective function are best explained by 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 initial similarity values. 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 keyword weight induced similarity part corresponds to updating similarity values between keywords in proportion to weights of the keywords to the respective documents they are contained in and the similarity between the documents. keyword weight induced similarity part also helps updating similarity values

between documents in proportion to weights of keywords they contain and the similarity between the contained keywords.

Minimizing the similarity induced keyword part corresponds to updating keyword weights in proportion to the following

- Similarity between v_1 and other keywords $v_2 \in G_2$
- Keyword weight of v_2 to documents $u_2 \in G_1$
- Similarity between u_1 and u_2

Therefore, even if frequency of a keyword such as "'Machine Translation"' in a publication is not high, it can achieve a high keyword weight if it contains many other similar keywords such as "'Bilingual Corpora"' and "'Word alignment"'.

5 Efficient Algorithm

We seek to minimize the objective function using Alternating Optimization (AO) (Bezdek and Hathaway, 2002), an approximate optimization method. Alternating optimization is an iterative procedure for minimizing (or maximizing) the function $f(x) = f(X_1, X_2, \ldots, X_t)$ jointly over all variables by alternating restricted minimizations over the individual subsets of variables $X_1, \ldots, X_t$.

In this optimization method, we partition the set of variables into a set of mutually exclusive, exhaustive subsets. We iteratively perform minimizations over each subset of variables while maintaining the other subsets of variables fixed. Formally, let the set of real-valued variables be $X = \{X_1, X_2, \ldots, X_N\}$ be partitioned into m subsets, $\{Y_1, Y_2, \ldots, Y_m\}$. Also, let $s_i = |Y_i|$. Then we begin with the initial set of values $\{Y_1^0, Y_2{}^0, \ldots, Y_m{}^0\}$ and make restricted minimizations of the following form,

$$\min_{Y_i \in \mathbb{R}^{s_i}} \{f(\underline{Y_1}^{r+1}, \ldots, \underline{Y_{i-1}}^{r+1}, Y_i, \underline{Y_{i+1}}^{r}, \ldots, \underline{Y_m}^{r})\} \tag{6}$$

where $i = 1, 2, \ldots, m$. The ***underline notation*** $\underline{Y_j}$ indicates that the subset of variables Y_j are fixed with respect to Y_i. In the context of our problem, we update each edge weight while maintaining other edge weights to be a constant. Then the problem boils down to the minimization problem over a single edge weight. For example, let us solve the minimization problem for edge weight corresponding to (u_i, v_j) where $u_i, v_j \in G_1$ (The case where $u_i, v_j \in G_2$ is analogous). Clearly the objective function is a convex function in $w_(u_i, v_j)$. The partial derivative of the objective function with respect to the edge weight is given below,

$$\begin{aligned}\frac{\partial \Omega(w,Z)}{\partial w_{u_i,v_j}} &= 2\alpha_0(w_{u_i,v_j} - w^*_{u_i,v_j}) \\ &+2\alpha_2 * \sum_{u_2,v_2 \in G_2} (w_{u_i,v_j} - \underline{w_{u_2,v_2}})\underline{Z_{u_1,u_2}}\underline{Z_{v_j,v_2}} \\ &+\alpha_3 * \sum_{u_2,v_2 \in G_2} (\underline{Z_{u_i,u_2}} - \underline{Z_{v_j,v_2}})^2 w_{u_i,v_j}\underline{w_{u_2,v_2}}\end{aligned} \tag{7}$$

.

To minimize the above function, we set the partial derivative to zero which gives us the following expression,

$$\begin{aligned}w_{u_j,v_k} = \quad & \frac{1}{C_1}(\alpha_0 w^*_{u_i,v_j} + \\ & \alpha_2 \sum_{u_2,v_2 \in G_2} \underline{Z_{u_i,u_2}}\, \underline{w_{u_2,v_2}}\, \underline{Z_{v_j,v_2}})\end{aligned} \tag{8}$$

where

$$\begin{aligned}C_1 = \quad & \alpha_0 + \alpha_2 \sum_{u_2,v_2 \in G_2} \underline{Z_{u_i,u_2}}\, \underline{Z_{v_j,v_2}} \\ & +\frac{\alpha_3}{2} \sum_{u_2,v_2 \in G_2} (\underline{Z_{u_i,u_2}} - \underline{Z_{v_j,v_2}})^2 \underline{w_{u_2,v_2}}\end{aligned}$$

Similarly, we can derive the update equation for keyword weights, Z_{u_i,u_j} as below,

$$\begin{aligned}Z_{u_i,u_j} \quad & = \frac{1}{C_2}(\alpha_1 Z^*_{u_i,u_j} + \\ & \alpha_3 \sum_{v_1 \in G_1} \sum_{v_2 \in G_2} \underline{w_{u_i,v_1}}\, \underline{w_{u_j,v_2}}\, \underline{Z_{v_1,v_2}})\end{aligned} \tag{9}$$

where,

$$\begin{aligned}C_2 = \quad & \alpha_1 + \alpha_3 \sum_{v_1 \in G_1} \sum_{v_2 \in G_2} \underline{w_{u_i,v_1}}\, \underline{w_{u_j,v_2}} \\ & +\frac{\alpha_2}{2} \sum_{v_1 \in G_1} \sum_{v_2 \in G_2} (\underline{w_{u_i,v_1}} - \underline{w_{u_j,v_2}})^2 \underline{Z_{v_1,v_2}}\end{aligned}$$

The similarity score between two nodes is proportional to the similarity between nodes in the other layer. For example, the similarity between two documents (keywords) is proportional to the similarity between the keywords the documents they contain (the documents they are contained in). C plays the role of a normalization constant. Therefore, for similarity between two nodes to be high, it is required that they not only contain a lot of similar nodes in the other graph but the similar nodes need to be important to the two target nodes.

Similarly, a particular keyword will have a high weight to a document if similar keywords have high weights to similar documents. Also, it is necessary that the similarity between the keywords and the documents are high.

It can be shown that equations 8 and 9 converge $q-linearly$ since the minimization problem is convex in each of the variables individually and hence has a global and unique minimizer (Bezdek and Hathaway, 2002).

5.1 Layered Random Walk Interpretation

The above algorithm has a very nice intuitive interpreation in terms of random walks over the two different graphs. Assume the initial weights are transition probability values after the graphs are normalized so that each row of the adjacency matrices sums to 1. Then the similarity between two nodes u and v in the same graph is computed as sum of two parts. The first part is α_0 times the initial similarity. This is necessary so that the optimized similarity values are not too far away from the initial similarity values. The second part corresponds to the probability of a random walk of length 3 starting at u and reaching v through two intermediate nodes from the other graph.

The semantics of the random walk depends on whether u, v are documents or keywords. For example, if the two nodes are documents, then the similarity between two documents d_1 and d_2 is the probability of random walk starting at document d_1 and then moving to a keyword k_1 and then moving to keyword k_2 and then to document d_2. Note that keywords k_1 and k_2 can be the same keyword which accounts for similarity between documents because they contain the same keyword.

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 graph, whose length is greater than 1. This is due to the iterative nature of the algorithm. For example, keywords "'Machine Translation"' and "'Bilingual corpora"' occur often together and hence any co-occurrence based similarity measure will assign a high initial similarity value. Hence two publications which contain these words will be assigned a non-zero similarity value after a single iteration. Also, "'Bilingual corpora"' and "'SMT"' (abbreviation for Statistical Machine Translation) can have a high initial similarity value which enables assiging a high similarity value between "'Machine Translation"' and "'SMT"'. This leads to a chain effect as the number of iterations increases which helps assign non-zero similarity values between semantically similar documents even if they do not contain common keywords.

6 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. There are three natural variants of the algorithm,

- *Unified*: We compare against the edge-weight regularization algorithm proposed in (Muthukrishnan et al., 2010). The algorithm has the same representation as our algorithm but the optimization is strictly defined over the edge weights in the two layers of the graph, $w_ij's$ and not on the keyword weights. Therefore, Z_ij are maintained constant throughout the algorithm.

- *Unified-binary*: In this variant, we initialize the keyword weights to 1, i.e, $Z_ij = 1$ whenever document i contains the keyword j.

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 1: Details of a few sample papers classified according to research topic

- *Unified-TFIDF*: We initialize the keyword weights to the TFIDF scores, Z_ij is set to the TFIDF score of keyword j for document i.

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 link similarity, we use the node similarity algorithm proposed by (Harel and Koren, 2001) using a random walk of length 3 on the link graph.

- *Linear combination*: The content similarity (CS) and link similarity (LS) between documents x and y are combined in a linear fashion as $\alpha CS(x, y) + (1 - \alpha)LS(x, y)$. We tried different values of α and report only the best accuracy that can be achieved using linear combination of similarity measures. Note that this is an upper bound on the accuracy of Multiple Kernel Learning with the restriction of the combination being affine.

We also compare our algorithm against the following algorithms *SC-MV*: We compare our algorithm against the spectral classification algorithm for data with multiple views (Zhou and Burges, 2007). 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. 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

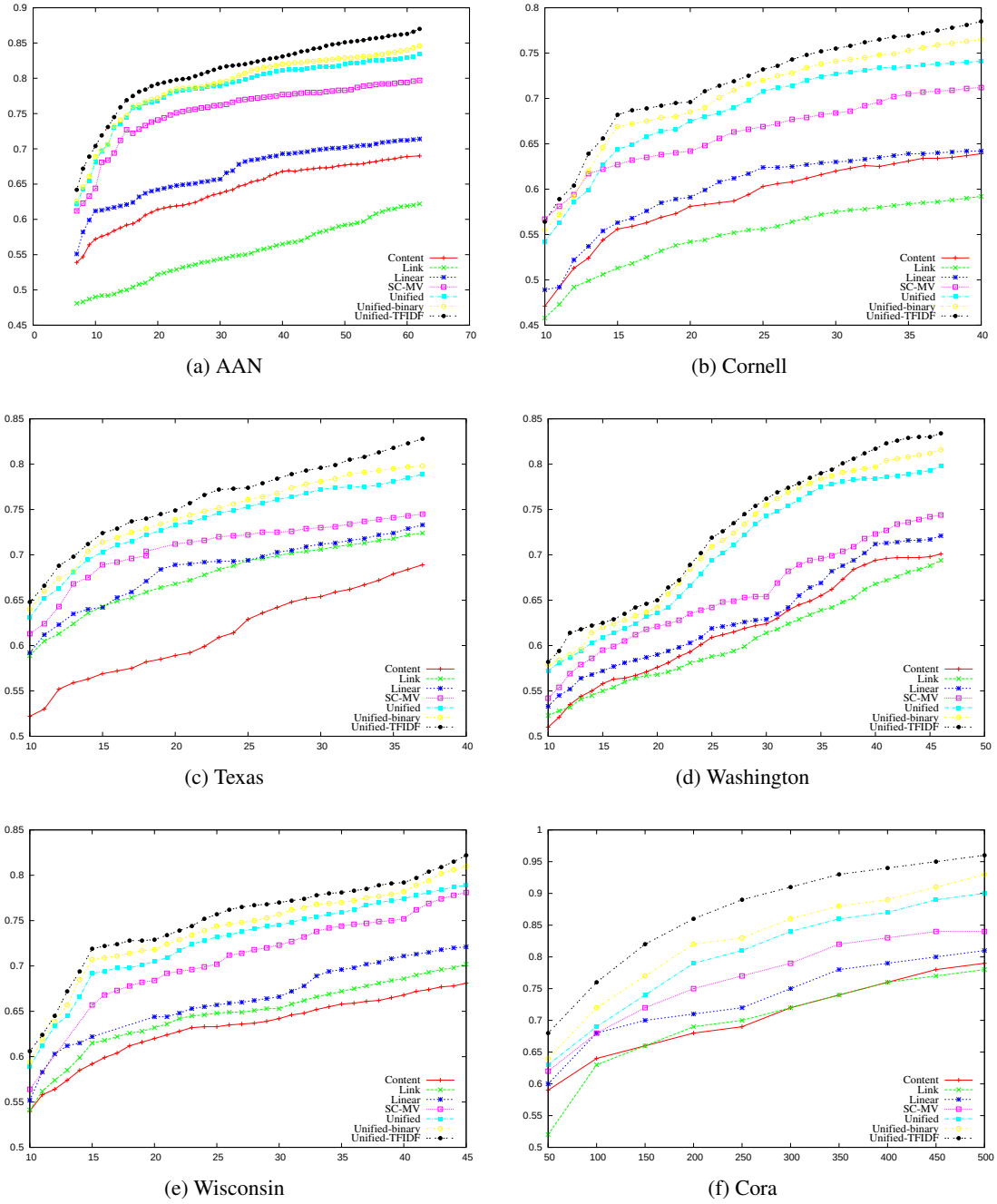
We use a semi-supervised graph classification algorithm (Zhu et al., 2003) to perform the classification.

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 (Dhillon et al., 2007) to perform the clustering.

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), (Radev et al., 2009) 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 (Ma-



(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.

chine 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 1 lists a few sample papers from each class.

WebKB(Sen et al., 2008): 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(Sen et al., 2008): 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.

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 keywords 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 documents and is initialized to the similarity between the documents due to the link structure. The link similarity between two documents is computed using the similarity measure proposed by (Harel and Koren, 2001) on the citation graph. We also performed experiments by initializing the similarity between documents to the keyword similarity. Although, our algorithm still outperforms other algorithms and the baselines (not shown due to space restrictions), the accuracy using citation similarity is higher.

7 Results and Discussion

Figure 1 shows the accuracy of the classification obtained using different similarity measures. It can be seen that the proposed algorithm (both the variants) performs much better than other similarity measures by a large margin. The algorithm performs much better when more information is provided in the form of TF-IDF scores. 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 instead of the normal feature vector representation. Thus, we can leverage on the similarity between the keywords and the objects (documents) to iteratively improve similarity in both layers. Whereas, in the case of the algorithm in (Zhou and Burges, 2007) all the graphs are isolated homogeneous graphs. Hence there is no information transfer across the different graphs.

For the clustering task, we use Normalized Mutual Information (NMI) (Strehl and Ghosh, 2002) between the ground truth clusters and the outputted clustering as the measure of clustering accuracy.

Table 2 shows the Normalized Mutual Information scores obtained by the different similarity measures on the different data sets.

8 Conclusion

In this paper, we have proposed a novel approach to compute similarity between documents and keywords iteratively. We formalized the problem of similarity estimation as an optimization problem induced by a regularization framework over edges in multiple graphs. We propose an efficient, iterative algorithm based on Alternating Optimization (AO) which has a neat, intuitive interpretation in terms of random walks over multiple graphs. We demonstrated the improved performance of the proposed algorithm over many different baselines and a state-of-the-art classifcation algorithm and a similarity measure which uses the same information as given to our algorithm.

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
Unified Similarity-Binary	0.80	0.68	0.56	0.69	0.74	0.66
Unified Similarity-TFIDF	**0.84**	**0.70**	**0.60**	**0.72**	**0.78**	**0.70**

Table 2: Normalized Mutual Information scores of the different similarity measures on the different data sets

References

Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. 2004. Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 6–, New York, NY, USA. ACM.

James Bezdek and Richard Hathaway. 2002. Some notes on alternating optimization. In Nikhil Pal and Michio Sugeno, editors, *Advances in Soft Computing AFSS 2002*, volume 2275 of *Lecture Notes in Computer Science*, pages 187–195. Springer Berlin.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*.

Corinna Cortes, Mehryar. Mohri, and Afshin Rostamizadeh. 2009. Learning non-linear combinations of kernels. In *In NIPS*.

Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. 2003. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 89–98, New York, NY, USA. ACM.

Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. 2007. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1944–1957, November.

William A. Gale and Kenneth Ward Church. 1991. A program for aligning sentences in bilingual corpora. In *In Proceedings of ACL*.

David Harel and Yehuda Koren. 2001. On clustering using random walks. In *Foundations of Software Technology and Theoretical Computer Science 2245*, pages 18–41. Springer-Verlag.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *In Proceedings of EMNLP*.

Pradeep Muthukrishnan, Dragomir Radev, and Qiaozhu Mei. 2010. Edge weight regularization over multiple graphs for similarity learning. In *In ICDM*.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, HLT-NAACL–Demonstrations '04, pages 38–41, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dragomir R. Radev, Pradeep Muthukrishnan, and Vahed Qazvinian. 2009. The ACL Anthology Network corpus. In *In Proceedings of the ACL Workshop on Natural Language Processing and Information Retrieval for Digital Libraries*.

Prithviraj Sen, Galileo Mark Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Magazine*, 29(3):93–106.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles: a knowledge reuse framework for combining partitionings. In *Eighteenth national conference on Artificial intelligence*, pages 93–98, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Dengyong Zhou and Christopher J. C. Burges. 2007. Spectral clustering and transductive learning with multiple views. In *ICML '07*, pages 1159–1166, New York, NY, USA.

Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003*, pages 912–919.