

On Text Clustering with Side Information

Charu C. Aggarwal ^{*1}, Yuchen Zhao ^{#2}, Philip S. Yu ^{#3}

**IBM T. J. Watson Research Center
Hawthorne, NY 10532*

¹ charu@us.ibm.com

*#University of Illinois at Chicago
Chicago, IL, USA*

² yzhao@cs.uic.edu

³ psyu@cs.uic.edu

Abstract—Text clustering has become an increasingly important problem in recent years because of the tremendous amount of unstructured data which is available in various forms in online forums such as the web, social networks, and other information networks. In most cases, the data is not purely available in text form. A lot of side-information is available along with the text documents. Such side-information may be of different kinds, such as the links in the document, user-access behavior from web logs, or other non-textual attributes which are embedded into the text document. Such attributes may contain a tremendous amount of information for clustering purposes. However, the relative importance of this side-information may be difficult to estimate, especially when some of the information is noisy. In such cases, it can be risky to incorporate side-information into the clustering process, because it can either improve the quality of the representation for clustering, or can add noise to the process. Therefore, we need a principled way to perform the clustering process, so as to maximize the advantages from using this side information. In this paper, we design an algorithm which combines classical partitioning algorithms with probabilistic models in order to create an effective clustering approach. We present experimental results on a number of real data sets in order to illustrate the advantages of using such an approach.

I. INTRODUCTION

The problem of text clustering arises in the context of many application domains such as the web, social networks, and other digital collections. The rapidly increasing amounts of text data in the context of these large online collections has led to an interest in creating scalable and effective mining algorithms. A tremendous amount of work has been done in recent years on the problem of clustering in text collections [4], [9], [24], [26], [33] in the database and information retrieval communities. However, this work is primarily designed for the problem of pure text clustering, in the absence of other kinds of attributes. In many application domains, a tremendous amount of side-information is also associated along with the documents. This is because text documents typically occur in the context of a variety of applications in which there may be a large amount of other kinds of database attributes or meta-information which may be useful to the clustering process. Some examples of such side-information are as follows:

- In an application in which we track user access behavior of web documents, the user-access behavior may be captured in the form of web logs. For each document,

the meta-information may correspond to the browsing behavior of the different users. Such logs can be used to enhance the quality of clustering in a way which is more meaningful to the user, and also application-sensitive. This is because the logs can often pick up subtle correlations in content, which cannot be picked up by the raw text alone.

- Many text documents contain links among them, which can also be treated as attributes. Such links contain a lot of useful information for clustering purposes. As in the previous case, such attributes may often provide insights about the correlations among documents in a way which may not be easily accessible from raw content.
- Many web documents have meta-data associated with them which correspond to different kinds of attributes such as the provenance or other information about the origin of the document. In other cases data such as ownership, location, or even temporal information may be informative for clustering purposes. In a number of network and user-sharing applications, documents may be associated with user-tags, which may also be quite informative.

While such side-information can sometimes be useful in improving the quality of the clustering process, it can be a risky approach when the side-information is noisy. In such cases, it can actually worsen the quality of the clustering. Therefore, we will use an approach which carefully ascertains the coherence of the clustering characteristics of the side information with that of the text content. This helps in magnifying the clustering effects of both kinds of data. The core of the approach is to determine a clustering in which the text attributes and side-information provide similar hints about the nature of the underlying clusters, and at the same time ignore those aspects in which conflicting hints are provided.

In order to achieve this goal, we will combine a partitioning approach with a probabilistic estimation process, which determines the coherence of the side-attributes in the clustering process. A probabilistic model on the side information uses the partitioning information (from text attributes) for the purpose of estimating the coherence of different clusters with side attributes. This helps in abstracting out the noise in the

membership behavior of different attributes. The partitioning approach is specifically designed to be very efficient for large data sets. This can be important in scenarios in which the data sets are very large. We will present experimental results on a number of real data sets, and illustrate the effectiveness and efficiency of the approach.

This paper is organized as follows. The remainder of this section will present the related work on the topic. In the next section, we will formalize the problem of text clustering with side information. We will also present an algorithm for the clustering process. In section 3, we will present the experimental results. Section 4 contains the conclusions and summary.

A. Related Work

The problem of text-clustering has been studied widely by the database community [15], [22], [30]. The major focus of this work has been on scalable clustering of multi-dimensional data of different types [15], [16], [22], [30]. A general survey of clustering algorithms may be found in [18]. The problem of clustering has also been studied quite extensively in the context of text-data. A survey of text clustering methods may be found in [3]. One of the most well known techniques for text-clustering is the scatter-gather technique [9], which uses a combination of agglomerative and partitional clustering. Other related methods for text-clustering which use similar methods are discussed in [24], [25]. Co-clustering methods for text data are proposed in [10], [11]. An Expectation Maximization (EM) method for text clustering has been proposed in [19]. Matrix-factorization techniques for text clustering are proposed in [28]. This technique selects words from the document based on their relevance to the clustering process, and uses an iterative EM method in order to refine the clusters. A closely related area is that of topic-modeling, event tracking, and text-categorization [5], [7], [12], [13]. In this context, a method for topic-driven clustering for text data has been proposed in [31]. Methods for text clustering in the context of keyword extraction are discussed in [14]. A number of practical tools for text clustering may be found in [20]. A comparative study of different clustering methods may be found in [26].

The problem of text clustering has also been studied in context of scalability in [4], [17], [33]. However, all of these methods are designed for the case of pure text data, and do not work for cases in which the text-data is combined with other forms of data. Some limited work has been done on clustering text in the context of network-based linkage information [1], [2], [6], [8], [21], [27], [32], [29], though this work is not applicable to the case of general side-information attributes. In this paper, we will provide a first approach to using other kinds of attributes in conjunction with text clustering. We will show the advantages of using such an approach over pure text-based clustering. Such an approach is especially useful, when the auxiliary information is highly informative, and provides effective guidance in creating more coherent clusters.

II. CLUSTERING WITH SIDE INFORMATION

In this section, we will discuss an approach for clustering text data with side information. We assume that we have a corpus \mathcal{S} of text documents. The total number of documents is N , and they are denoted by $T_1 \dots T_N$. It is assumed that the set of distinct words in the entire corpus \mathcal{S} is denoted by \mathcal{W} . Associated with each document T_i , we have a set of side attributes \overline{X}_i . Each set of side attributes \overline{X}_i has d dimensions, which are denoted by $(x_{i1} \dots x_{id})$. We refer to such attributes as *auxiliary* attributes. For ease in notation and analysis, we assume that each side-attribute x_{id} is binary, though both numerical and categorical attributes can easily be converted to this format in a fairly straightforward way. This is because the different values of the categorical attribute can be assumed to be separate binary attributes, whereas numerical data can be discretized to binary values with the use of attribute ranges. Some examples of such side-attributes are as follows:

- In a web log analysis application, we assume that x_{ir} corresponds to the 0-1 variable, which indicates whether or not the i th document has been accessed by the r th user. This information can be used in order to cluster the web pages in a site in a more informative way than a techniques which is based purely on the content of the documents. As in the previous case, the number of pages in a site may be large, but the number of documents accessed by a particular user may be relatively small.
- In a network application, we assume that x_{ir} corresponds to the 0-1 variable corresponding to whether or not the i th document T_i has a hyperlink to the r th page T_r . If desired, it can be implicitly assumed that each page links to itself in order to maximize linkage-based connectivity effects during the clustering process. Since hyperlink graphs are large and sparse, it follows that the number of such auxiliary variables are high, but only a small fraction of them take on the value of 1.
- In a document application with associated GPS or provenance information, the possible attributes may be drawn on a large number of possibilities. Such attributes will naturally satisfy the sparsity property.

As noted in the examples above, such auxiliary attributes are quite sparse in many real applications. This can be a challenge from an efficiency perspective, unless the sparsity is carefully taken into account during the clustering process. Therefore, our techniques will be designed to account for such sparsity.

We note that our technique is not restricted to binary auxiliary attributes, but can also be applied to attributes of other types. When the auxiliary attributes are of other types (quantitative or categorical), they can be converted to binary attributes with the use of a simple transformation process. For example, numerical data can be discretized into binary attributes. Even in this case, the derived binary attributes are quite sparse especially when the numerical ranges are discretized into a large number of attributes. In the case of categorical data, we can define a binary attribute for each possible categorical value. In many cases, the number of such

values may be quite large. Therefore, we will design our techniques under the implicit assumption that such attributes are quite sparse. The formulation for the problem of clustering with side information is as follows:

Text Clustering with Side Information: *Given a corpus S of documents denoted by $T_1 \dots T_N$, and a set of auxiliary variables \overline{X}_i associated with document T_i , determine a clustering of the documents into k clusters which are denoted by $C_1 \dots C_k$, based on both the text content and the auxiliary variables.*

We will use the auxiliary information in order to provide additional insights, which can improve the quality of clustering. In many cases, such auxiliary information may be noisy, and may not have useful information for the clustering process. Therefore, we will design our approach in order to magnify the coherence between the text content and the side-information, when this is detected. In cases, in which the text content and side-information do not show coherent behavior for the clustering process, the effects of those portions of the side-information are marginalized.

A. The COATES Algorithm

In this section, we will describe our algorithm for text clustering with side-information. We refer to this algorithm as *COATES* throughout the paper, which corresponds to the fact that it is a *CO*ntent and *A*uxiliary attribute based *TE*xt *cl*ustering algorithm. We assume that an input to the algorithm is the number of clusters k . As in the case of all text-clustering algorithms, it is assumed that stop-words have been removed, and stemming has been performed in order to improve the discriminatory power of the attributes. The algorithm requires two phases:

- **Initialization:** We use a lightweight initialization phase in which a standard text clustering approach is used without any side-information. For this purpose, we use the algorithm described in [24]. The centroids and the partitioning created by the clusters formed in the first phase provide an initial starting point for the second phase. We note that the first phase is based on text only, and does not use the auxiliary information.
- **Main Phase:** The main phase of the algorithm is executed after the first phase. This phase starts off with these initial groups, and iteratively reconstructs these clusters with the use of *both* the text content and the auxiliary information. This phase performs alternating iterations which use the text content and auxiliary attribute information in order to improve the quality of the clustering. We call these iterations as *content* iterations and *auxiliary iterations* respectively. The combination of the two iterations is referred to as a *major iteration*. Each major iteration thus contains *two minor iterations*, corresponding to the auxiliary and text-based methods respectively.

The focus of the first phase is simply to construct an initialization, which provides a good starting point for the clustering process based on text content. Since the key techniques for content and auxiliary information integration are in the second

phase, we will focus most of our subsequent discussion on the second phase of the algorithm. The first phase is simply a direct application of the text clustering algorithm proposed in [24]. The overall approach uses alternating minor iterations of content-based and auxiliary attribute-based clustering. These phases are referred to as *content-based* and *auxiliary attribute-based* iterations respectively. The algorithm maintains a set of seed centroids, which are subsequently refined in the different iterations. In each content-based phase, we assign a document to its closest seed centroid based on a text similarity function. The centroids for the k clusters created during this phase are denoted by $L_1 \dots L_k$. Specifically, the cosine similarity function is used for assignment purposes. In each auxiliary phase, we create a probabilistic model, which relates the attribute probabilities to the cluster-membership probabilities, based on the clusters which have already been created in the most recent text-based phase. The goal of this modeling is to examine the coherence of the text clustering with the side-information attributes. Before discussing the auxiliary iteration in more detail, we will first introduce some notations and definitions which help in explaining the clustering model for combining auxiliary and text variables.

We assume that the k clusters associated with the data are denoted by $C_1 \dots C_k$. In order to construct a probabilistic model of membership of the data points to clusters, we assume that each auxiliary iteration has a *prior* probability of assignment of documents to clusters (based on the execution of the algorithm so far), and a *posterior* probability of assignment of documents to clusters with the use of auxiliary variables in that iteration. We denote the prior probability that the document T_i belongs to the cluster C_j by $P(T_i \in C_j)$. Once the pure-text clustering phase has been executed, the *a-priori* cluster membership probabilities of the auxiliary attributes are generated with the use of the last content-based iteration from this phase. The *a-priori* value of $P(T_i \in C_j)$ is simply the fraction of documents which have been assigned to the cluster C_j . In order to compute the *posterior* probabilities $P(T_i \in C_j | \overline{X}_i)$ of membership of a record at the end of the auxiliary iteration, we use the auxiliary attributes \overline{X}_i which are associated with T_i . Therefore, we would like to compute the conditional probability $P(T_i \in C_j | \overline{X}_i)$. We will make the approximation of considering only those auxiliary attributes (for a particular document), which take on the value of 1. Since we are focussing on sparse binary data, the value of 1 for an attribute is a much more informative event than the default value of 0. Therefore, it suffices to condition only on the case of attribute values taking on the value of 1. For example, let us consider an application in which the auxiliary information corresponds to users which are browsing specific web pages. In such a case, the clustering behavior is influenced much more significantly by the case when a user *does* browse a particular page, rather than one in which the user *does not* browse a particular page, because most pages will typically not be browsed by a particular user. This is generally the case across many sparse data domains such as attributes corresponding to links, discretized numeric data, or categorical data which is

quite often of very high cardinality (such as zip codes).

Furthermore, in order to ensure the robustness of the approach, we need to eliminate the noisy attributes. This is especially important, when the number of auxiliary attributes is quite large. Therefore, at the beginning of each auxiliary iteration, we compute the *gini-index* of each attribute based on the clusters created by the last content-based iteration. This gini-index provides a quantification of the discriminatory power of each attribute with respect to the clustering process. The gini-index is computed as follows. Let f_{rj} be the fraction of the records in the cluster \mathcal{C}_j (created in the last content-based iteration), for which the attribute r takes on the value of 1. Then, we compute the *relative presence* p_{rj} of the attribute r in cluster j as follows:

$$p_{rj} = \frac{f_{rj}}{\sum_{m=1}^k f_{rm}} \quad (1)$$

The values of p_{rj} are defined, so that they sum to 1 over a particular attribute r and different clusters j . We note that when all values of p_{rj} take on a similar value of $1/k$, then the attribute values are evenly distributed across the different clusters. Such an attribute is not very discriminative with respect to the clustering process, and it should not be used for clustering. While the auxiliary attributes may have a different clustering behavior than the textual attributes, it is also expected that informative auxiliary attributes are at least somewhat related to the clustering behavior of the textual attributes. This is generally true of many applications such as those in which auxiliary attributes are defined either by linkage-based patterns or by user behavior. On the other hand, completely noisy attributes are unlikely to have any relationship to the text content, and will not be very effective for mining purposes. Therefore, we would like the values of p_{rj} to vary across the different clusters. We refer to this variation as *skew*. The level of skew can be quantified with the use of the gini-index. The gini-index of attribute r is denoted by \mathcal{G}_r , and is defined as follows:

$$\mathcal{G}_r = \sum_{j=1}^k p_{rj}^2 \quad (2)$$

The value of \mathcal{G}_r lies between $1/k$ and 1. The more discriminative the attribute, the higher the value of \mathcal{G}_r . In each iteration, we use only the auxiliary attributes for which the gini-index is above a particular threshold γ . The value of γ is picked to be 1.5 standard deviations below the mean value of the gini-index in that particular iteration. We note that since the clusters may change from one iteration to the next, and the gini-index is defined with respect to the current clusters, the values of the gini-index will also change over the different iterations. Therefore, different auxiliary attributes may be used over different iterations in the clustering process, as the quality of the clusters become more refined, and the corresponding discriminative power of auxiliary attributes can also be computed more effectively.

Let R_i be a set containing the indices of the attributes in \overline{X}_i which are considered discriminative for the clustering

Algorithm COATES(NumClusters: k , Corpus: $T_1 \dots T_N$, Auxiliary Attributes: $\overline{X}_1 \dots \overline{X}_N$);

begin

Use content-based algorithm in [24] to create initial set of k clusters $\mathcal{C}_1 \dots \mathcal{C}_k$;

Let centroids of $\mathcal{C}_1 \dots \mathcal{C}_k$ be denoted by $L_1 \dots L_k$;

$t = 1$;

while not(*termination_criterion*) **do**

begin

{ First minor iteration }

Use cosine-similarity of each document T_i to centroids $L_1 \dots L_k$ in order to determine the closest cluster to T_i and update the cluster assignments $\mathcal{C}_1 \dots \mathcal{C}_k$;

Denote assigned cluster index for document T_i by $q_c(i, t)$;

Update cluster centroids $L_1 \dots L_k$ to the centroids of updated clusters $\mathcal{C}_1 \dots \mathcal{C}_k$;

{ Second Minor Iteration }

Compute gini-index of \mathcal{G}_r for each auxiliary attribute r with respect to current clusters $\mathcal{C}_1 \dots \mathcal{C}_k$;

Mark attributes with gini-index which is γ standard-deviations below the mean as non-discriminatory;

{ for document T_i let R_i be the set of attributes which take on the value of 1, and for which gini-index is discriminatory; }

for each document T_i use the method discussed in section 2 to determine the posterior probability $P^n(T_i \in \mathcal{C}_j | R_i)$;

Denote $q_a(i, t)$ as the cluster-index with highest posterior probability of assignment for document T_i ;

Update cluster-centroids $L_1 \dots L_k$ with the use of posterior probabilities as discussed in section 2;

$t = t + 1$;

end

end

Fig. 1. The COATES Algorithm

process, and for which the value of the corresponding attribute is 1. For example, let us consider an application in which we have 1000 different auxiliary attributes. If the dimension indices of the attributes in the vector \overline{X}_i which take on the value of 1 are 7, 120, 311, and 902 respectively, then we have $R_i = \{7, 120, 311, 902\}$. Therefore, instead of computing $P(T_i \in \mathcal{C}_j | \overline{X}_i)$, we will compute the conditional probability of membership based on a particular value of the set R_i . We define this quantity as the *attribute subset based conditional probability of cluster membership*.

Definition 1: The attribute-subset based conditional probability of cluster membership of the document T_i to the cluster \mathcal{C}_j is defined as the conditional probability of membership of document T_i to cluster \mathcal{C}_j based only on the set of attributes R_i from \overline{X}_i which take on the value of 1.

The attribute-subset based conditional probability of document T_i to cluster \mathcal{C}_j is denoted by $P^s(T_i \in \mathcal{C}_j | R_i)$. We note that if R_i contains a modest number of attributes, then it is hard to directly approximate $P^s(T_i \in \mathcal{C}_j | R_i)$ in a data-driven manner. We note that these are the *posterior* probabilities of cluster membership *after* the auxiliary-attribute based iteration, given that particular sets of auxiliary attribute values

are presented in the different documents. We will use these posterior probabilities in order to re-adjust the cluster centroids during the auxiliary attribute-based iterations. The *a-priori* probabilities of membership are assigned based on the current *content-based* iteration of the clustering, and are denoted by $P^a(T_i \in C_j)$. The a-priori and a-posteriori probabilities are related as follows:

$$P^s(T_i \in C_j | R_i) = P^a(R_i | T_i \in C_j) \cdot P^a(T_i \in C_j) / P^a(R_i) \quad (3)$$

The above equation follows from the simple expansion of the expression for the conditional probabilities. We will also see that each expression on the right-hand side can be estimated in a *data-driven* manner with the use of the conditional estimates from the last iteration. Specifically, the value of $P^a(T_i \in C_j)$ is simply the fraction of the documents assigned to the cluster C_j in the last content-based (minor) iteration.

In order to evaluate posterior probabilities from Equation 3, we also need to estimate $P^a(R_i)$ and $P^a(R_i | T_i \in C_j)$. Since R_i may contain many attributes, this is essentially a joint probability, which is hard to estimate exactly with a limited amount of data. Therefore, we make a *naive Bayes approximation* in order to estimate the values of $P^a(R_i)$ and $P^a(R_i | T_i \in C_j)$. In order to make this approximation, we assume that the different attributes of R_i are independent of one another. Then, we can approximate $P(R_i)$ as follows:

$$P^a(R_i) = \prod_{r \in R_i} P^a(x_{ir} = 1) \quad (4)$$

This value of $P^a(x_{ir} = 1)$ is simply the fraction of the documents in which the value of the attribute x_{ir} is one. This can be easily estimated from the underlying document collection. Similarly, we use the independence assumption in order to estimate the value of $P^a(R_i | T_i \in C_j)$.

$$P^a(R_i | T_i \in C_j) = \prod_{r \in R_i} P^a(x_{ir} = 1 | T_i \in C_j) \quad (5)$$

The value of $P^a(x_{ir} = 1 | T_i \in C_j)$ is the fraction of the documents **in cluster** C_j in which the value of attribute x_{ir} is one. This value can be estimated from the last set of clusters obtained from the content-based clustering phase. For each cluster C_j , we determine the fraction of records for which the value of r th auxiliary attribute is 1. We note that the only difference between the second definition and the first is the fact that in the second case, we are computing the fraction of the documents in the cluster j for which the r th attribute is 1. Then, we can substitute the results of Equations 4 and 5 in Equation 3 in order to obtain the following:

$$P^s(T_i \in C_j | R_i) = P^a(T_i \in C_j) \cdot \prod_{r \in R_i} \frac{P^a(x_{ir} = 1 | T_i \in C_j)}{P^a(x_{ir} = 1)} \quad (6)$$

In order to simplify the expression further in order to make it more intuitive, we compute the interest ratio $I(r, j)$ as the expression $\frac{P^a(x_{ir}=1|T_i \in C_j)}{P^a(x_{ir}=1)}$. This is the interest ratio for the r th auxiliary attribute in relation to the j th cluster C_j . We formally define the interest ratio as follows:

Definition 2: The interest ratio $I(r, j)$ defines the importance of attribute r with respect to the j th cluster C_j . This is defined as the ratio of the probability of a document belonging to cluster C_j , when the r th auxiliary attribute of the document is set to 1, to the ratio of the same probability unconditionally. Intuitively, the ratio indicates the proportional increase in likelihood of a document belonging to a particular cluster (from the current set of clusters) because of a particular auxiliary attribute value. In other words, we have:

$$I(r, j) = \frac{P^a(x_{ir} = 1 | T_i \in C_j)}{P^a(x_{ir} = 1)} \quad (7)$$

A value of $I(r, j)$, which is significantly greater than 1 indicates that the auxiliary attribute r is highly related to the current cluster C_j . The break-even value of $I(r, j)$ is one, and it indicates that the auxiliary attribute r is not specially related to the cluster C_j . We can further simplify the expression for the conditional probability in Equation 6 as follows:

$$P^s(T_i \in C_j | R_i) = P^a(T_i \in C_j) \cdot \prod_{r \in R_i} I(r, j) \quad (8)$$

We note that the values of $P^s(T_i \in C_j | R_i)$ should sum to 1 over the different values of (cluster index) j . However, this may not be the case in practice, because of the use of the independence approximation while computing the probabilities for the different attributes. Therefore, we *normalize* the posteriori probabilities to $P^n(T_i \in C_j | R_i)$, so that they add up to 1, as follows:

$$P^n(T_i \in C_j | R_i) = \frac{P^s(T_i \in C_j | R_i)}{\sum_{m=1}^k P^s(T_i \in C_m | R_i)} \quad (9)$$

These normalized posterior probabilities are then used in order to re-adjust the centroids $L_1 \dots L_k$. Specifically, each document T_i is assigned to the corresponding centroid L_j with a probability proportional to $P^n(T_i \in C_j | R_i)$, and the text content of the document T_i is added to the cluster centroid L_j of C_j with a scaling factor proportional to $P^n(T_i \in C_j | R_i)$. We note that such an iteration supervises the text content of the centroids on the basis of the auxiliary information, which in turn affects the assignment of documents to centroids in the next content-based iteration. Thus, this approach iteratively tries to build a consensus between the text content-based and auxiliary attribute-based assignments of documents to clusters.

In the next *content-based* iteration, we assign the documents to the modified cluster-centroids based on the cosine similarity of the documents to the cluster centroids [23]. Each document is assigned to its closest cluster centroid based on the cosine similarity. The assigned documents are then aggregated in order to create a new centroid meta-document which aggregates the frequency of the words in the documents for that cluster. The least frequent words in this cluster are then pruned away, so as to use a vector of only the most frequent words in the cluster centroid with their corresponding frequencies. This new assignment of documents to clusters is again used for defining the a-priori probabilities for the auxiliary attributes in the next iteration. A key issue for the algorithm is the convergence of the algorithm towards a uniform solution. In order to compute

convergence, we assume that we have an identifier associated with each cluster in the data. This identifier does not change from one iteration to the next for a particular centroid. Within the t th major iteration, we compute the following quantities for each document for the two different minor iterations:

- We compute the cluster identifier to which the document T_i was assigned in the content-based step of the t th major iteration. This is denoted by $qc(i, t)$.
- We compute the cluster identifier to which the document T_i had the highest *probability of assignment* in the auxiliary-attribute set of the t th major iteration. This is denoted by $qa(i, t)$.

In order to determine when the iterative process should terminate, we would like the documents to have assignments to similar clusters in the $(t - 1)$ th and t th steps at the end of both the auxiliary-attribute and content-based steps. In other words, we would like $qc(i, t - 1)$, $qa(i, t - 1)$, $qc(i, t)$ and $qa(i, t)$ to be as similar as possible. Therefore, we compute the number of documents for which all four of these quantities are the same. As the algorithm progresses, the number of such records will initially increase rapidly, and then slowly level off. The algorithm is assumed to have terminated, when the number of such records does not increase by more than 1% from one iteration to the next. At this point, the algorithm is assumed to have reached a certain level of stability in terms of its assignment behavior, and it can therefore be terminated. An important point to be remembered is that the output to the algorithm are both the vectors $qc(\cdot, t)$ and $qa(\cdot, t)$. While the clustering process is inherently designed to converge to clusters which use both content and auxiliary information, some of the documents cannot be made to agree in the clustering behavior with the use of different criteria. The common assignments in $qc(\cdot, t)$ and $qa(\cdot, t)$ correspond to the cases in which the content-based and auxiliary-assignments can be made to agree with a well-designed clustering process, and the differences in the two vectors correspond to those documents which show different clustering behavior for the auxiliary and content-based information. Such documents are interesting, because they provide *intensional understanding* of how some of the content-based information may be different from the auxiliary information in the clustering process. The overall clustering algorithm is illustrated in Figure 1.

B. Smoothing Issues

An important smoothing issue arises in process of evaluating the right hand side of Equation 6. Specifically, the expression denoted by $\prod_{r \in R_i} \frac{P^a(x_{ir}=1|T_i \in \mathcal{C}_j)}{P^a(x_{ir}=1)}$ may contain zero values for $P^a(x_{ir} = 1|T_i \in \mathcal{C}_j)$. Even a single such zero value could set the whole expression to zero. This will result in an ineffective clustering process. In order to avoid this problem, we use a smoothing parameter ϵ_r for the r th auxiliary attribute. Specifically, the expression in Equation 6. Therefore, the corresponding expression is evaluated by $\prod_{r \in R_i} \frac{P^a(x_{ir}=1|T_i \in \mathcal{C}_j) + \epsilon_r}{P^a(x_{ir}=1) + \epsilon_r}$. The value of ϵ_r is fixed to within a small fraction of $P^a(x_{ir} = 1)$.

III. EXPERIMENTAL RESULTS

In this section, we compare our method against a number of baseline techniques on real and synthetic data sets. We refer to our approach as *Content and Auxiliary attribute based TExt cluStering (COATES)*. As the baseline, we used two different methods: **(1)** An efficient projection based clustering approach [24] which adapts the k -means approach to text. This approach is widely known to provide excellent clustering results in a very efficient way. We refer to this algorithms as *SchutzeSilverstein [text only]* in all figure legends in the experimental section. **(2)** We adapt the k -means approach with the use of both text and side information directly. We refer to this baseline as *K-Means [text+side]* in all figure legends.

Thus, we compare our algorithms with baselines which are chosen in such a way that we can evaluate the advantage of our approach over both a pure text-clustering method and a natural alternative which uses both text and side-information. We will show that our approach has significant advantages over both.

A. Data Sets

We used three real data sets in order to test our approach. The data sets used were as follows:

(1) Cora Data Set: The Cora data set¹ contains 19,396 scientific publications in the computer science domain. Each research paper in the Cora data set is classified into a topic hierarchy. On the leaf level, there are 73 classes in total. We used the second level labels in the topic hierarchy, and there are 10 class labels, which are *Information Retrieval, Databases, Artificial Intelligence, Encryption and Compression, Operating Systems, Networking, Hardware and Architecture, Data Structures Algorithms and Theory, Programming and Human Computer Interaction*. We further obtained two types of side information from the data set: citation and authorship. These were used as separate attributes in order to assist in the clustering process. There are 75,021 citations and 24,961 authors. One paper has 2.58 authors in average, and there are 50,080 paper-author pairs in total.

(2) DBLP-Four-Area Data Set: The *DBLP-Four-Area data set* [27] is a subset extracted from DBLP that contains four data mining related research areas, which are database, data mining, information retrieval and machine learning. This data set contains 28,702 authors, and the texts are the important terms associated with the papers that were published by these authors. In addition, the data set contained information about the conferences in which each author published. There are 20 conferences in these four areas and 44,748 author-conference pairs. Besides the author-conference attribute, we also used co-authorship as another type of side information, and there were 66,832 coauthor pairs in total.

(3) IMDB Data Set: The **Internet Movie DataBase (IMDB)** is an online collection² of movie information. We obtained ten-year movie data from 1996 to 2005 from IMDB in order to

¹<http://www.cs.umass.edu/~mccallum/code-data.html>

²<http://www.imdb.com>

perform text clustering. We used the plots of each movie as text to perform pure text clustering. The genre of each movie is regarded as its class label. We extracted movies from the top four genres in *IMDB* which were labeled by *Short*, *Drama*, *Comedy*, and *Documentary*. We removed the movies which contain more than two above genres. There were 9,793 movies in total, which contain 1,718 movies from the *Short* genre, 3,359 movies from the *Drama* genre, 2,324 movies from the *Comedy* genre and 2,392 movies from the *Documentary* genre. The names of the directors, actors, actresses, and producers were used as categorical attributed corresponding to side information. The *IMDB* data set contained 14,374 movie-director pairs, 154,340 movie-actor pairs, 86,465 movie-actress pairs and 36,925 movie-producer pairs.

B. Evaluation Metrics

The aim is to show that our approach is superior to natural clustering alternatives with the use of either pure text or with the use of both text and side information. In each data set, the class labels were given, but they were not used in the clustering process. For each class, we computed the cluster purity, which is defined as the fraction of documents in the clusters which correspond to its dominant class. The average cluster purity over all clusters (weighted by cluster size) was reported as a surrogate for the quality of the clustering process. Let the number of data points in the k clusters be denoted by $n_1 \dots n_k$. We denote the dominant input cluster label in the k clusters by $l_1 \dots l_k$. Let the number of data points with input cluster label l_i be denoted by c_i . Then, the overall cluster purity P is defined by the fraction of data points in the clustering which occur as a dominant input cluster label. Therefore, we have:

$$P = \frac{\sum_{i=1}^k c_i}{\sum_{i=1}^k n_i} \quad (10)$$

The cluster purity always lies between 0 and 1. Clearly, a perfect clustering will provide a cluster purity of almost 1, whereas a poor clustering will provide very low values of the cluster purity. For efficiency, we tested the execution time of our method with respect to the baseline over three real data sets.

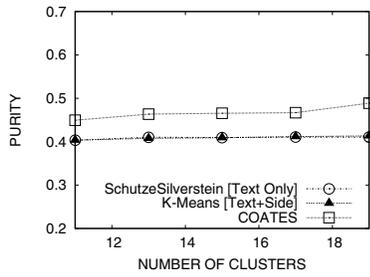
Since the quality of content-based clustering varies based on random initialization, we repeated each test ten times with different random seeds and reported the average as the final score. Unless otherwise mentioned, the default value of the number of input clusters used for the *Cora*, *DBLP-Four-Area* and *IMDB* data sets were 16, 8 and 6 respectively. These default sizes were chosen based on the size of the underlying data set. All results were tested on a Debian GNU/Linux server. The system used double dual-core 2.4 GHz Opteron processors with 4GB RAM.

C. Effectiveness Results

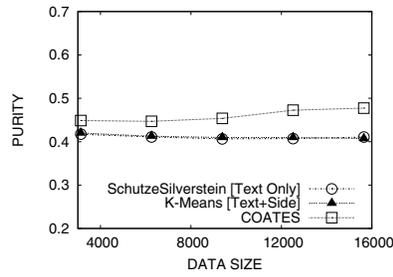
In this section, we will present the effectiveness results for all three data sets. The effectiveness results for the two baseline algorithms and *COATES* algorithms with increasing number of clusters for the *CORA*, *DBLP* and *IMDB* data sets

are illustrated in Figures 2(a), (c) and (e) respectively. The number of clusters is illustrated on the X -axis, whereas the cluster purity with respect to the ground truth is illustrated on the Y -axis. We notice that the purity will slightly increase when the number of clusters increases on all three data sets. This is quite natural, since larger number of clusters results in a finer granularity of partitioning. We further notice that *COATES* outperforms the baselines on all three data sets by a wide margin in terms of purity. In the *Cora* and *IMDB* data sets, there was not much difference between the two baselines (which differ in terms of their use of side information or not), because the k -means baseline was unable to leverage the low frequency side information effectively, and the effects of the text content dominated the clustering results. This shows that the side-information is not only important in the clustering process, but it is also important to leverage it properly in order to obtain the maximum gains during the clustering. The improvement in quality over both baselines remains competitive throughout different settings on the number of clusters. This suggests that the *COATES* algorithm is consistently superior to the different baselines, irrespective of the value of this parameter. This suggests that the *COATES* scheme is able to use the side information in a robust and consistent way over different numbers of clusters.

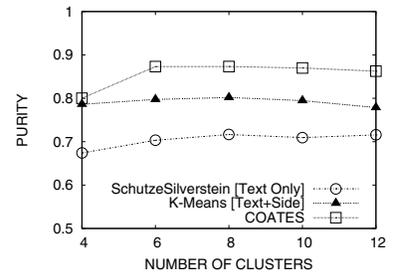
We also tested the effectiveness of the method with increasing data size. This is done by sampling a portion of the data, and reporting the results for different sample sizes. The effectiveness results with increasing data size for the *Cora*, *DBLP*, and *IMDB* data sets are illustrated in Figures 2(b), (d) and (f) respectively. In all figures, the X -axis illustrates the size of data and the Y -axis represents the cluster purity. We find that the improvement in quality with the use of side information is more pronounced for the case of larger data sets. In particular, we can see that at the lower end of the X -axis (corresponding to data samples which are only about 20% of the full size), the effectiveness of adding side information is insignificant in terms of improvement. That is because of the fact that the sparse side information associated with data sets becomes less effective when the data set itself is very small. In such cases, most side information attributes appear very infrequently and thus, it is difficult to leverage them in order to obtain better clustering assignments. When the data size increases to around 40% of the whole data, it has enough side information to improve the quality of clustering. As the size of the data set increases, the improvement in quality becomes even more significant. We can also infer that *COATES* works especially well when the data set is fairly large, because in such cases there are lots of repetitions of the same attribute value of a particular kind of side information, especially when it is sparse. The repetition of the same attribute value of the side information is critical in being able to leverage it for improving clustering quality. In addition the *COATES* algorithm was consistently superior to both the baselines in terms of clustering quality.



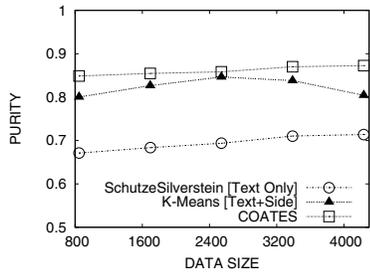
(a) CORA (Effectiveness vs. number of clusters)



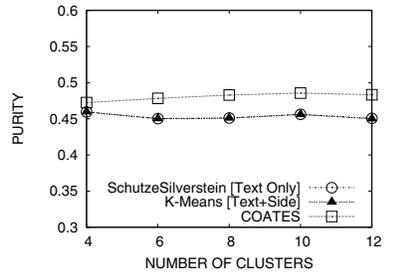
(b) CORA (Effectiveness vs. data size)



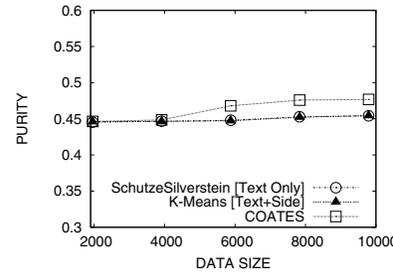
(c) DBLP (Effectiveness vs. number of clusters)



(d) DBLP (Effectiveness vs. data size)

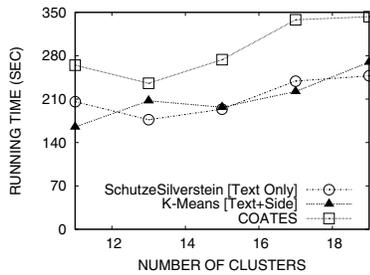


(e) IMDB (Effectiveness vs. number of clusters)

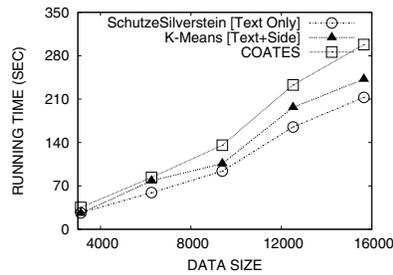


(f) IMDB (Effectiveness vs. data size)

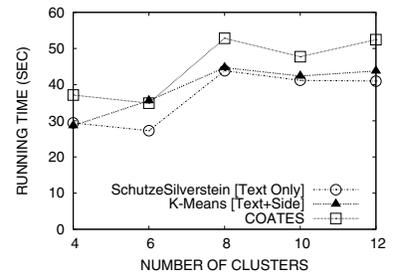
Fig. 2. Effectiveness Results on all Data Sets



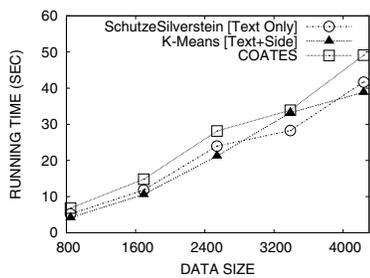
(a) CORA (Efficiency vs. number of clusters)



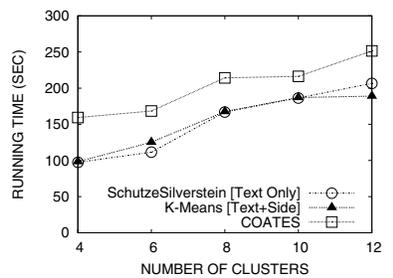
(b) CORA (Efficiency vs. data size)



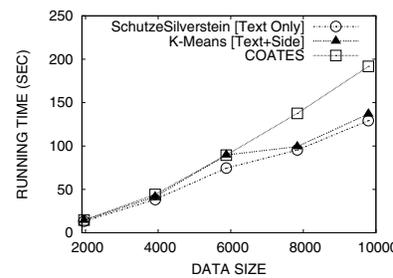
(c) DBLP (Efficiency vs. number of clusters)



(d) DBLP (Efficiency vs. data size)

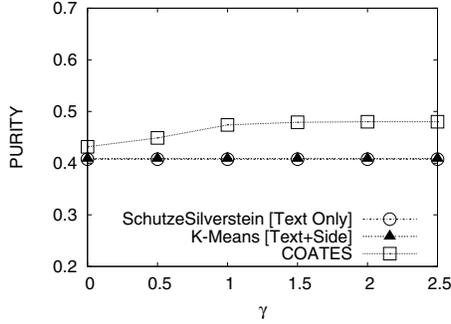


(e) IMDB (Efficiency vs. number of clusters)

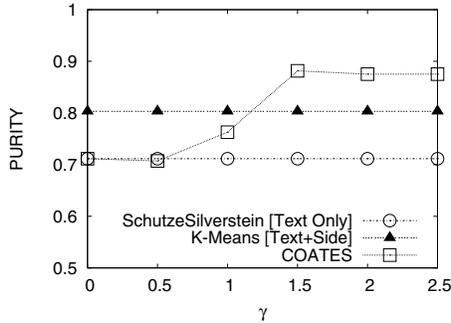


(f) IMDB (Efficiency vs. data size)

Fig. 3. Efficiency Results on all Data Sets

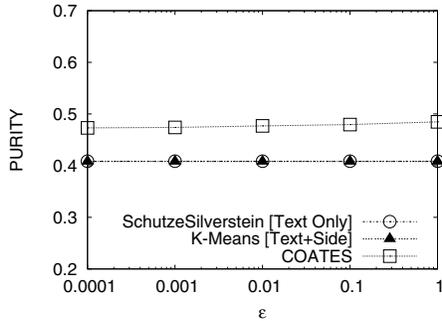


(a) Cora Data Set

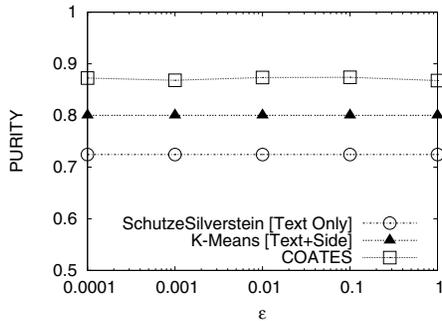


(b) DBLP-Four-Area Data Set

Fig. 4. Sensitivity Analysis with Threshold γ



(a) Cora Data Set



(b) DBLP-Four-Area Data Set

Fig. 5. Sensitivity Analysis with Smoothing Parameter ϵ

D. Efficiency Results

In this section, we present the running times for *COATES* and the two different baselines. We note that all approaches use the same text preprocessing methods, such as stops words removal, stemming and term frequency computation, and the time required for side-information pre-processing is negligible. Therefore, the pre-processing time for both methods is virtually the same for all methods, and we present the running times only for the clustering portions in order to sharpen the comparisons and make them more meaningful. We further note that since the *COATES* algorithm is expected to incorporate side-information into the clustering process in a much more significant way than both the baselines, its running times are correspondingly expected to be somewhat higher. The goal is to show that the overheads associated with the better qualitative results of the *COATES* algorithm are somewhat low. We first tested the efficiency of the different methods with respect to the number of clusters, and the results of the *CORA*, *DBLP* and *IMDB* data sets are reported in Figures 3(a), (c) and (e) respectively. The number of clusters are illustrated on the *X*-axis, and the running time is illustrated on the *Y*-axis. From the figures, it is evident that the *COATES* algorithm consumes more running time compared with the baselines, though it is only slightly slower than both baselines. The reason is that *COATES* method focusses on the side information in a much more focussed way, which slightly increases its running time. For example, in the *Cora* data set, the side information comprises 50,080 paper-author pairs and 75,021 citing paper-cited paper pairs. It is important to understand that the *COATES* clustering process needs to create a separate iteration in order to effectively process the side information. Therefore, the slight overhead of 10%- to 30% on the running time of the *COATES* algorithm is quite acceptable. From the figures, it is evident that the *COATES* algorithm scales linearly with increasing number of clusters for all data sets. This is because the distance function computations and posterior probability computations scale linearly with increasing number of clusters.

It is also valuable to test the scalability of the proposed approach with increasing data size. The results for the *Cora*, *DBLP*, and *IMDB* data sets are illustrated in Figures 3(b), (d) and (f) respectively. In all figures, the *X*-axis illustrates the size of data, and the *Y*-axis illustrates the running time. As in the previous case, the *COATES* algorithm consumes a little more time than the baseline algorithms. These results also show that the running time scales linearly with increasing data size. This is because the number of distance function computations and posterior probability computations scale linearly with the number of documents in each iteration. The linear scalability implies that the technique can be used very effectively for large text data sets.

E. Sensitivity Analysis

We also tested the sensitivity of the *COATES* algorithm with respect to two important parameters. We will present the sensitivity results on only the *Cora* and *DBLP-Four-Area data sets* because of space limitations.

As mentioned in the algorithm in Figure 1, we used threshold γ to select discriminative auxiliary attributes. While the default value of the parameter was chosen to be 1.5, we also present the effects of varying this parameter. Figures 4 (a) and (b) show the effects on cluster purity by varying the value of threshold γ from 0 to 2.5 on the *Cora* and *DBLP-Four-Area data sets*. In both figures, the *X*-axis illustrates the value of threshold γ , and the *Y*-axis presents the purity. The results are constant for both baseline methods because they do not use this parameter. It is evident from both figures that setting the threshold γ too low results in purity degradation, since the algorithm will pick a lot of noisy auxiliary attributes in this case. On both data sets, the *COATES* algorithm achieves good purity results when γ is set to be 1.5. Further increasing the value of γ will reduce the purity slightly because setting γ too high will result in also missing essential attributes. However, the algorithm shows good performance for a fairly large range of values of γ . This suggests that the approach is quite robust.

Figures 5 (a) and (b) illustrate the impact of the smoothing parameter ϵ on the effectiveness of the clustering process. The value of the smoothing parameter ϵ is illustrated on a logarithmic scale on the *X*-axis, whereas the purity is illustrated on the *Y*-axis. Although the *X*-axis is on a logarithmic scale, the purity is quite stable across all ranges. But we have to note that ϵ should not be set to too large a value, because this will reduce the positive effects of the additional knowledge added by the auxiliary variables. However, the robustness of the approach over large ranges of values of ϵ shows that the method can be used quite safely for different values of the parameter ϵ .

IV. CONCLUSIONS AND SUMMARY

In this paper, we presented a method for text clustering with the use of side-information. Many forms of text-databases contain a large amount of side-information or meta-information, which may be used in order to improve the clustering process. In order to design the clustering method, we combined an iterative partitioning technique with a probability estimation process which computes the importance of different kinds of side-information. We present results on real data sets illustrating the effectiveness of our approach. The results show that the use of side-information can greatly enhance the quality of text clustering, while maintaining a high level of efficiency.

ACKNOWLEDGEMENTS

The research of the first author was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice hereon. The research of the second and third authors was supported in part by NSF through grants CNS-1115234, IIS 0905215,

IIS-0914934, and DBI-0960443, and Google Mobile 2014 Program.

REFERENCES

- [1] C. C. Aggarwal and H. Wang, *Managing and Mining Graph Data*, Springer, 2010.
- [2] C. C. Aggarwal, *Social Network Data Analytics*, Springer, 2011.
- [3] C. C. Aggarwal and C.-X. Zhai, *Mining Text Data*, Springer, 2012.
- [4] C. C. Aggarwal and P. S. Yu, "A Framework for Clustering Massive Text and Categorical Data Streams," in *SIAM Conf. on Data Mining*, pp. 477–481, 2006.
- [5] C. C. Aggarwal, S. C. Gates, and P. S. Yu, "On using partial supervision for text categorization," in *IEEE TKDE*, vol. 16(2), pp. 245–255, 2006.
- [6] R. Angelova and S. Siersdorfer, "A neighborhood-based approach for clustering of linked document collections," in *CIKM Conf.*, pp. 778–779, 2006.
- [7] A. Banerjee and S. Basu, "Topic models over text streams: A study of batch and online unsupervised learning," in *SDM Conf.*, pp. 437–442, 2007.
- [8] J. Chang and D. Blei, "Relational Topic Models for Document Networks," in *AISTATIS*, pp. 81–88, 2009.
- [9] D. Cutting, D. Karger, J. Pedersen, and J. Tukey, "Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections," in *ACM SIGIR Conf.*, pp. 318–329, 1992.
- [10] I. Dhillon, "Co-clustering Documents and Words using bipartite spectral graph partitioning," in *ACM KDD Conf.*, pp. 269–274, 2001.
- [11] I. Dhillon, S. Mallela and D. Modha, "Information-theoretic Co-Clustering," in *ACM KDD Conf.*, pp. 89–98, 2003.
- [12] M. Franz, T. Ward, J. S. McCarley, and W. J. Zhu, "Unsupervised and supervised clustering for topic tracking," in *ACM SIGIR Conf.*, pp. 310–317, 2001.
- [13] G. P. C. Fung, J. X. Yu, and H. Lu, "Classifying text streams in the presence of concept drifts," in *PAKDD Conf.*, pp. 373–383, 2004.
- [14] H. Frigui and O. Nasraoui, "Simultaneous clustering and dynamic keyword weighting for text documents, Survey of text mining," Michael Berry, Ed, *Springer*, pp. 45–70, 2004.
- [15] S. Guha, R. Rastogi, and K. Shim, "CURE: An Efficient Clustering Algorithm for Large Databases," in *ACM SIGMOD Conf.*, pp. 73–84, 1998.
- [16] S. Guha, R. Rastogi, and K. Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes," in *Inf. Syst.*, vol. 25(5), pp. 345–366, 2000.
- [17] Q. He, K. Chang, E.-P. Lim and J. Zhang, "Bursty feature representation for clustering text streams," in *SDM Conf.*, pp. 491–496, 2007.
- [18] A. Jain and R. Dubes, *Algorithms for clustering data*, Prentice-Hall, Inc., 1988.
- [19] T. Liu, S. Liu, Z. Chen, and W.-Y. Ma, "An evaluation of feature selection for text clustering," in *ICML Conf.*, pp. 488–495, 2003.
- [20] A. McCallum, "Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering," <http://www.cs.cmu.edu/mccallum/bow>, 1996.
- [21] Q. Mei, D. Cai, D. Zhang and C.-X. Zhai, "Topic Modeling with Network Regularization," in *WWW Conf.*, pp. 101–110, 2008.
- [22] R. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," in *VLDB Conf.*, pp. 144–155, 1994.
- [23] G. Salton, *An Introduction to Modern Information Retrieval*, McGraw Hill, 1983.
- [24] H. Schutze and C. Silverstein, "Projections for Efficient Document Clustering," in *ACM SIGIR Conf.*, pp. 74–81, 1997.
- [25] C. Silverstein and J. Pedersen, "Almost-constant time clustering of arbitrary corpus sets," in *ACM SIGIR Conf.*, pp. 60–66, 1997.
- [26] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *Text Mining Workshop, KDD*, pp. 109–110, 2000.
- [27] Y. Sun, J. Han, J. Gao, and Y. Yu, "iTopicModel: Information Network Integrated Topic Modeling," in *ICDM Conf.*, pp. 493–502, 2009.
- [28] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," in *ACM SIGIR Conf.*, pp. 267–273, 2003.
- [29] T. Yang, R. Jin, Y. Chi, and S. Zhu, "Combining link and content for community detection: a discriminative approach," in *ACM KDD Conf.*, pp. 927–936, 2009.

- [30] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," *ACM SIGMOD Conf.*, pp. 103–114, 1996.
- [31] Y. Zhao and G. Karypis, "Topic-driven clustering for document datasets," in *SIAM Conf. on Data Mining*, pp. 358–369, 2005.
- [32] Y. Zhou, H. Cheng and J. X. Yu, "Graph clustering based on structural/attribute similarities," *PVLDB*, vol. 2(1), pp. 718–729, 2009.
- [33] S. Zhong, "Efficient Streaming Text Clustering," *Neural Networks*, vol 18, Issue 5–6, pp. 790–798, 2005.