

Collaborative Document Clustering

Khaled Hammouda*

Mohamed Kamel†

Abstract

Document clustering has been traditionally studied as a centralized process. There are scenarios when centralized clustering does not serve the required purpose; *e.g.* documents spanning multiple digital libraries need not be clustered in one location, but rather clustered at each location, then enriched by receiving more information from other locations. A distributed collaborative approach for document clustering is proposed in this paper. The main objective here is to allow peers in a network to form independent opinions of local document grouping, followed by exchange of cluster summaries in the form of keyphrase vectors. The nodes then expand and enrich their local solution by receiving recommended documents from their peers based on the peer judgement of the similarity of local documents to the exchanged cluster summaries. Results show improvement in final clustering after merging peer recommendations. The approach allows independent nodes to achieve better local clustering by having access to distributed data without the cost of centralized clustering, while maintaining the initial local clustering structure and coherency.

1 Introduction

Information is better utilized when it is processed to be easier to find, better organized, or summarized for easier digestion. Areas dealing with such problems are at the cross-roads of information retrieval, machine learning (*e.g.* classification and clustering), and statistical analysis. Text and web mining problems in particular use methodologies often spanning those areas.

Document clustering is an area that deals with the unsupervised grouping of text documents into meaningful groups, usually representing topics in the document collection. It is one way to organize information without requiring prior knowledge about the classification of documents, and could be used as a base for document

categorization by forming an initial classification.

Document clustering has many applications, such as clustering of search engine results to present organized and understandable results to the user (*e.g.* Vivisimo¹), clustering documents in a collection (*e.g.* digital libraries), automated (or semi-automated) creation of document taxonomies (*e.g.* Yahoo and Open Directory styles), and efficient information retrieval by focusing on relevant subsets (clusters) rather than whole collections. Perhaps the most popular application of document clustering is the Google News² service, which uses document clustering techniques to group news articles from multiple news sources to provide a combined overview of news around the Web.

Traditionally, document clustering has been studied as a centralized process; *i.e.* all data is assumed to be present at a central site, then a single process applies clustering to the data. A more wider view of how clustering can be applied in distributed environments is outlined in Table 1.

Table 1: Types of data and clustering process distribution

	Centralized Data	Distributed Data
Centralized Clustering	CD-CC	DD-CC
Distributed Clustering	CD-DC	DD-DC

Centralized Data - Centralized Clustering (CD-CC)

This is the standard approach where the clustering process and data both reside on the same machine.

Distributed Data - Centralized Clustering (DD-CC)

Data might be dispersed across different machines, a typical case in the Web domain, while the clustering process runs on a single machine.

Centralized Data - Distributed Clustering (CD-DC)

Data is stored in one location, with clustering processes running on different machines accessing the same data; a typical case of *parallel processing*.

Distributed Data - Distributed Clustering (DD-DC)

The highest level of distribution, where both the data and the clustering processes are distributed.

*Systems Design Engineering, Pattern Analysis and Machine Intelligence Research Group, University of Waterloo, Waterloo, Ontario, Canada. Email: hammouda@pami.uwaterloo.ca

†Electrical & Computer Engineering, Pattern Analysis and Machine Intelligence Research Group, University of Waterloo, Waterloo, Ontario, Canada. Email: mkamel@pami.uwaterloo.ca

¹www.vivisimo.com

²news.google.com

The problem with centralized clustering is that sometimes raw data is distributed across different databases, making it either infeasible or impossible to apply centralized clustering. In other situations having a global clustering solution of the distributed data is not required; a local clustering solution is sufficient, which can be augmented or enhanced by having access to *summarized cluster information* from peer nodes.

To better motivate the above scenarios, consider a set of digital libraries (*e.g.* archived articles from online publishers). Each digital library can form an opinion about the topic groups found in its collection by applying local clustering. To enrich the local clustering solution, each library can receive recommendations from other libraries on what articles, which it currently does not carry, can fit into its current clustering solution. Thus achieving wider accessibility to previously unavailable data and expanding its clustering solution, while maintaining its initial structure which is based on the local data.

Another scenario is when access to peer data is restricted. In this case we do not have a full global dataset to operate upon. Instead, we can still achieve a local clustering that includes pointers to peer data. This is possible by allowing peers to look at local cluster summaries and then recommend to us those documents that they think can fit within our clustering.

In this paper we are presenting an approach that enables *collaborative clustering* among distributed nodes. By collaborative we mean the ability of one node to benefit other nodes based on their needs. The stricter term “cooperative clustering” is not being used as the aim of cooperative clustering is to achieve a common benefit (*e.g.* a global clustering solution). However, in the collaborative approach we are aiming at enriching the local clustering solution of each individual node based on recommendations from peer nodes.

The collaborative approach presented here can be considered one approach to “distributed clustering”. There are many approaches that rely on a distributed framework for doing the task; the collaborative approach tries to utilize the distributed framework to achieve better local clustering through peer involvement and recommendation.

Figure 1 illustrates the collaborative clustering system presented in this paper. From the point of view of system architecture, we are adopting a *peer-to-peer* framework. Each node in the network has access to part of the whole document collection. The network is a connected graph, in which every node is connected to every other node.

Every node starts by generating a local cluster model of the documents to which it has access. The

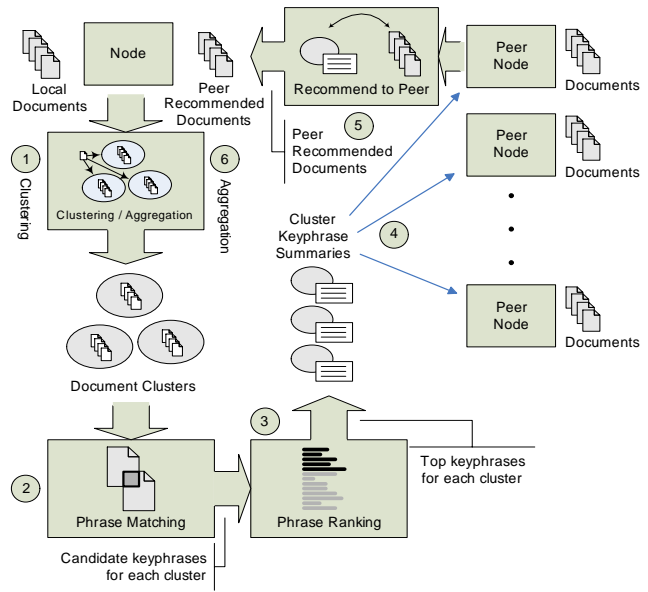


Figure 1: Collaborative Document Clustering System

goal is to enhance the clustering solution at each node by allowing peers to receive recommended documents so that the local clustering quality is maximized.

The information exchanged between nodes is minimized by forming of cluster summaries, represented as a vector of core keyphrases in each cluster. This step involves an elaborate multi-document keyphrase extraction algorithm that accurately extracts those keyphrases that best describe each cluster [10].

Each node applies a similarity calculation between received cluster summaries and its own set of documents. It then recommends to each peer what set of documents can benefit the peer’s clustering solution. Finally, the receiving peers decide which documents should be merged with their own clusters. Eventually, each node should have an improved set of clusters over the initial solution, facilitated by having access to global information from its peers.

The rest of this paper is organized as follows. Section 2 presents related work, and positions the work within the literature. Section 3 presents the collaborative document clustering model. Section 4 presents the algorithms for initial cluster generation, cluster summarization using keyphrase extraction, and collaborative clustering. Section 5 includes explanation and discussion of the experimental results. Finally, a conclusion and future research directions are outlined in Section 6.

2 Related Work

The literature is rich on data mining topics. Text mining in particular receives good attention due to its wide applicability in many domains. Clustering as a methodology in general has roots in the statistical analysis as well as the machine learning fields. However, we have found very little number of attempts in the area of distributed clustering, which is considered a fairly new area due to recent advances in networking and fast processors. In this section we try to relate the work presented here to other work in the literature, and position the work appropriately.

Data clustering is an established field. Jain *et al* [11, 12] cover the topic very well from the point of view of cluster analysis theory, and they break down the methodologies mainly into *partitional* and *hierarchical* clustering methods. In our work the clustering algorithm used falls under the partitional category, since we do not build hierarchies of clusters. However we allow clusters to overlap. Chakrabarti [5] also discusses various types of clustering methods and categorizes them into partitioning, geometric embedding, and probabilistic approaches. Since our algorithm is mainly dependent on statistical analysis, we can position it as a probabilistic algorithm.

Kosala and Blockeel [14] touch upon many aspects of web mining, showing the differences between web content mining, web structure mining, and web usage mining. They also discuss what kind of methodologies are used in web mining. Since we are analyzing the actual text content of web pages, as opposed to web link structure or server logs, we can say that our work falls under the category of web content mining.

Text mining research in general rely on a vector space model, first proposed in 1975 Salton *et al* [19, 18, 17] to model text documents as vectors in the feature space. Features are considered to be the words in the document collection, and feature values come from different term weighting schemes, the most popular of which is the TF-IDF (Term Frequency-Inverse Document Frequency) term weighting scheme. This model is simple but assumes independence between words in a document, which is not a major problem for statistical-based methods, but poses difficulty in phrase-based analysis. Thus, we rely on a model, the Document Index Graph, which explicitly represents phrases as a graph structure [9]. This model is the basis for efficient keyphrase extraction in our work.

Keyphrase extraction can be divided into keyphrase extraction from single documents, often treated as a supervised learning problem [20], and extraction from multiple documents, often treated as an unsupervised problem [6]. The work by Barzilay *et al* [1] seems

to have common approach to ours in terms of multi-document summarization. However, they focus on one domain, news articles, to detect the same event in multiple documents, and they use natural language processing techniques to supplement summaries with synthesized text. Again, most research in this area actually treats keyphrases as individual words, again breaking the phrase structure, which is a key in properly labeling clusters, as is used in this paper.

Posing the problem as a distributed data mining problem as opposed to the traditional centralized approach was largely due to Kargupta *et al* [13]. They proposed a distributed agent architecture in which each agent has a local model of the world. Agents then cooperate to achieve better global solution. In many aspects this is very related to our work. The main difference is that we do not cast the problem into a multi-agent paradigm, but rather as straightforward distributed processing problem. In addition, our method does not directly try to achieve a better global solution, but rather achieves this indirectly by maximizing the local solutions through collaboration of peers.

The work presented by Eisenhardt *et al* [7] seems to have very similar goal to ours: to solve the document clustering problem using a distributed peer-to-peer network. They use the k-means clustering algorithm, modified to work in a distributed fashion. The algorithm works by propagating the cluster mean vector to peers, which merge the mean vector with their own solution and report back the merged mean to the sending peer. They report improvement in speed up compared to centralized clustering, but they do not provide an evaluation of clustering quality in the distributed environment. A similar system can be found in [15], but the problem is posed from the information retrieval point of view.

3 Collaborative Document Clustering

The proposed framework consists of a number of models that describe different aspects of the problem. In this section we discuss how we represent the distributed nodes in the system, how to represent the data distribution, how to represent clusters of documents, and finally how the documents themselves are represented.

3.1 Node Distribution Model The node distribution model follows a peer-to-peer model. The network is represented as a connected graph $\mathbf{G}(\mathbf{N}, \mathbf{L})$

where \mathbf{N} : is the set of *nodes* $\{n_i\}$, $i = 1, \dots, N_N$.

\mathbf{L} : is the set of *links* $\{l_{ij}\}$ between every pair of nodes n_i and n_j , $i, j = 1, \dots, N_N$, $i \neq j$. The number of links in the network $N_L = N_N(N_N - 1)/2$.

This kind of connectedness is chosen for the sake of simplicity, since the typical scenarios for our approach involves only several nodes.

The number of nodes is static; *i.e.* N_N is fixed for a certain configuration and cannot be dynamically changed. We, however, test with different network sizes for different values of N_N .

Node links are assumed to be equally weighted. Nodes are assumed to have identical roles in the network, hence the peer-to-peer framework. The model also suggests that each node is assumed to have information about how to find all other nodes.

3.2 Data Distribution Model The global view of the data is a collection of documents $\mathbf{D} = \{\mathbf{d}_k\}$, $k = 1, \dots, N_D$. The original classification of the documents is assumed to be known, but not used during clustering. It is only used during evaluation.

The set of topics³ are represented as $\mathbf{T} = \{t_c\}$, $c = 1, \dots, N_T$. Each document is assumed to belong to one topic; *i.e.* $\text{Topic}(\mathbf{d}_k) \in \mathbf{T}$.

The document collection is assumed to be randomly, but evenly, distributed among the network nodes; *i.e.* each node holds the same number of documents, which is a percentage α of the total number of documents N_D . We refer to the parameter α as the *distribution ratio*. There could be overlap between the nodes in terms of the documents they hold; *i.e.* $1/N_N \leq \alpha \leq 1$. Thus, node i will hold a set of documents $\mathbf{D}_i \subseteq \mathbf{D}$, such that $|\mathbf{D}_i| = N_{D_i} = \alpha \cdot N_D$.

This type of document distribution mimics realistic scenarios, where each node can have access to either an exclusive part of the document collection, or there could be an overlap between the documents in each node.

3.3 Cluster Model Upon clustering the documents, each node will have created a set of document clusters that best fit its local document collection. Thus, each node n_i maintains a set of clusters $\mathbf{C}_i = \{\mathbf{c}_r\}$, $r = 1, \dots, N_{C_i}$. A cluster contains a subset of the documents in the node; *i.e.* \mathbf{c}_r contains $\mathbf{D}_r \subseteq \mathbf{D}_i$.

Clusters are allowed to overlap; *i.e.* each document can belong to more than one cluster, either in the same node or across different nodes. Thus, $\forall \mathbf{d}_k$, it is possible that $\exists \mathbf{c}_1, \mathbf{c}_2 \in \mathbf{C}$, such that $\mathbf{d}_k \in \mathbf{c}_1$ and $\mathbf{d}_k \in \mathbf{c}_2$.

Document-to-cluster membership is represented as a relation $R(\mathbf{D}, \mathbf{C})$, which is a binary membership matrix:

$$\mathcal{M} = [m_{k,r}]$$

$$m_{k,r} = \begin{cases} 1 & \text{iff } \mathbf{d}_k \in \mathbf{c}_r \\ 0 & \text{otherwise} \end{cases}$$

A projection of \mathcal{M} over the documents dimension yields the the number of documents in each cluster:

$$[\mathcal{M} \downarrow \mathbf{D}]_r = \sum_k m_{k,r}$$

Similarly, a projection of \mathcal{M} over the clusters dimension yields the the number of clusters to which each document belongs:

$$[\mathcal{M} \downarrow \mathbf{C}]_k = \sum_r m_{k,r}$$

In the process of collaborative clustering, there will be a need to represent cluster summaries in the form of keyphrase vectors, so that they can be exchanged between peers. The summary of cluster \mathbf{c}_r is represented as a keyphrase vector \mathbf{p}_r , and is referred to as the cluster *core*. The set of cluster cores corresponding to the set of clusters at node n_i is $\mathbf{P}_i = \{\mathbf{p}_r\}$, and will be referred to as the *node summary*. The keyphrases in each cluster core are the top k keyphrases extracted from the cluster using the algorithm described in Section 4.2.

3.4 Document Data Model Document features are represented as sentences rather than individual words, as is the case in most text mining approaches. The rationale for choosing phrase-based modeling is the inherent tendency of this model to make keyphrase extraction from document clusters straightforward (see section 4.2 for details).

The model assumes that the constituents of a document is a set of sentences, which in turn are composed of a set of terms. A document is represented as a vector of sentences, each of which is a vector of consecutive terms:

$$(3.1a) \quad \mathbf{d} = \{\mathbf{s}_i\},$$

$$(3.1b) \quad \mathbf{s}_i = (\mathbf{t}_i, w_i, l_i),$$

$$(3.1c) \quad \mathbf{t}_i = \{t_{ij}\},$$

$$(3.1d) \quad l_i = |\mathbf{t}_i|$$

\mathbf{s}_i : is *sentence* i in the document,

\mathbf{t}_i : is the *term* vector of sentence \mathbf{s}_i ,

t_{ij} : is the *term* j in sentence \mathbf{s}_i ,

w_i : is the *weight* associated with sentence \mathbf{s}_i , and

l_i : is the *length* of sentence \mathbf{s}_i .

Sentence weights are assigned according to their significance in the document (*e.g.* title, headings, *etc.*). A standard pre-processing is applied to the documents, in which stop words are removed, and different forms

³The term *topic* is used to refer to the document class to avoid confusion in mathematical notation.

of the same word are removed through stemming using the standard Porter algorithm [16].

4 Collaborative Document Clustering Algorithm

4.1 Initial Cluster Generation Each node starts by forming an local clustering solution, to be enriched later by collaborative clustering. A number of traditional clustering solutions were investigated and achieved comparable results. These included hierarchical, single-pass, k-nearest neighbor, and similarity histogram-based clustering (SHC) [9]. Hierarchical clustering usually performed slightly better than the other three, but since the collaborative clustering algorithm is a collaborative derivative of SHC, we chose it to do the initial clustering for the sake of consistency. The decision was also largely based on the fact that we are more interested in the improvement observed after merging peer recommendations to the local clustering solution, rather than in the quality of the initial clustering itself.

In SHC, the coherency of a cluster is represented as a **Cluster Similarity Histogram**.

Cluster Similarity Histogram: A concise statistical representation of the set of pairwise document similarities distribution in the cluster. A number of *bins* in the histogram correspond to fixed similarity value intervals. Each bin contains the count of pair-wise document similarities in the corresponding interval.

A coherent cluster should have high pairwise document similarities. Figure 2 shows a typical cluster similarity histogram, where the distribution is almost a normal distribution. A perfect cluster would have a histogram where the similarities are all maximum, while a loose cluster would have a histogram where the similarities are all minimum.

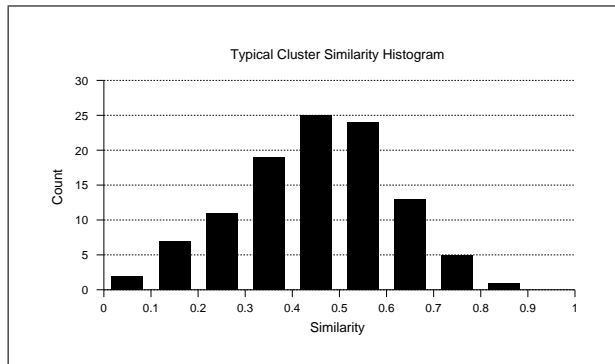


Figure 2: Cluster Similarity Histogram

We judge the quality of a similarity histogram

(cluster cohesiveness) by calculating the ratio of the count of similarities above a certain similarity threshold R_T to the total count of similarities. The higher this ratio, the more cohesive is the cluster. Let N_{D_c} be the number of the documents in a cluster. The number of pair-wise similarities in the cluster is $N_{R_c} = N_{D_c}(N_{D_c} + 1)/2$. Let $\mathbf{R} = \{r_i : i = 1, \dots, N_{R_c}\}$ be the set of similarities in the cluster. The histogram of the similarities in the cluster is represented as:

$$(4.2a) \quad \mathbf{H}_c = \{h_i : i = 1, \dots, B\}$$

$$(4.2b) \quad h_i = \text{count}(r_k) \quad r_{li} \leq r_k < r_{ui}$$

where B : the number of histogram bins,
 h_i : the count of similarities in bin i ,
 r_{li} : the lower similarity bound of bin i , and
 r_{ui} : the upper similarity bound of bin i .

The histogram ratio (HR) of a cluster is the measure of cohesiveness of the cluster as described above, and is calculated as:

$$(4.3a) \quad HR(\mathbf{c}) = \frac{\sum_{i=T}^B h_i}{\sum_{j=1}^B h_j}$$

$$(4.3b) \quad T = \lfloor R_T \cdot B \rfloor$$

where $HR(\mathbf{c})$: the histogram ratio of cluster \mathbf{c} ,
 R_T : the similarity threshold, and
 T : the bin number corresponding to the similarity threshold.

The algorithm works by maintaining high HR for each cluster. New documents are tested against each cluster, adding them to appropriate clusters if they do not degrade the HR of that cluster significantly. Provisions are also made so as not to allow a chain reaction of “bad” documents being added to the same cluster, thus bringing its cohesiveness down significantly.

The algorithm works incrementally by receiving a new document, and for each cluster calculates the cluster histogram before and after simulating the addition of the document (lines 4-6). The old and new histogram ratios are compared and if the new ratio is greater than or equal to the old one, the document is added to the cluster. If the new ratio is less than the old one by no more than ε and still above HR_{\min} , it is added (lines 7-9). Otherwise it is not added. If after checking all clusters the document was not assigned to any cluster, a new cluster is created and the document is added to it (lines 11-15).

4.2 Cluster Summarization Using Keyphrase Extraction

Once the initial clusters have been formed at each node, a process of cluster summarization is conducted. The summary of each cluster is represented

