Text Clustering with Seeds Affinity Propagation

Renchu Guan, Xiaohu Shi, Maurizio Marchese, Chen Yang, and Yanchun Liang

Abstract—Based on an effective clustering algorithm—Affinity Propagation (AP)—we present in this paper a novel semisupervised text clustering algorithm, called Seeds Affinity Propagation (SAP). There are two main contributions in our approach: 1) a new similarity metric that captures the structural information of texts, and 2) a novel seed construction method to improve the semisupervised clustering process. To study the performance of the new algorithm, we applied it to the benchmark data set Reuters-21578 and compared it to two state-of-the-art clustering algorithms, namely, k-means algorithm and the original AP algorithm. Furthermore, we have analyzed the individual impact of the two proposed contributions. Results show that the proposed similarity metric is more effective in text clustering (F-measures ca. 21 percent higher than in the AP algorithm) and the proposed semisupervised strategy achieves both better clustering results and faster convergence (using only 76 percent iterations of the original AP). The complete SAP algorithm obtains higher F-measure (ca. 40 percent improvement over k-means and AP) and lower entropy (ca. 28 percent decrease over k-means and AP), improves significantly clustering execution time (20 times faster) in respect that k-means, and provides enhanced robustness compared with all other methods.

Index Terms—Affinity propagation, text clustering, cofeature set, unilateral feature set, significant cofeature set.

1 Introduction

Clustering digital objects (e.g., text documents) by identifying a subset of representative examples plays an important role in recent text mining and information retrieval research. In fact, organizing a large amount of objects into meaningful clusters (clustering) is often used to browse a collection of objects and organize the results returned by a search engine [1], [2]. After the clustering process, the obtained clusters are represented with examples, which can include all or part of the features that appear in the cluster members. During cluster-based query processing, only those clusters that contain examples similar to the query are considered for further comparisons with cluster members, e.g., documents. This strategy, sometimes called Cluster-Based Retrieval, is intended to improve both efficiency and effectiveness of the document retrieval systems [3], [4], [5], [6]. Our work focuses on the proposal and detailed analysis of a new effective and fast clustering algorithm that can be used in cluster-based retrieval tasks.

Traditional approaches for clustering data are based on metric similarities, i.e., nonnegative, symmetric, and satisfying the triangle inequality measures. More recent approaches, like Affinity Propagation (AP) algorithm [7], can take as input also general nonmetric similarities. For instance, in the domain of image clustering, AP can use as input metric selected segments of images' pairs [8]. Accordingly, AP has been used to solve a wide range of clustering problems, such as image processing tasks [7], [8], gene detection tasks [9], and individual preferences predictions [10]. Affinity Propagation is derived as an application of the max-sum algorithm in a factor graph, i.e., it searches for the minima of an energy function on the basis of message passing between data points [7]. The clustering performance depends on the similarity measure and message updating frequency. For its simplicity, general applicability, and good performance, AP has already been used in text clustering. By using AP to preprocess texts, Ma et al. developed an incremental method [11] for text clustering. Wang et al. combined AP with a parallel strategy for e-learning resources clustering [12]. However, they used AP only as an unsupervised algorithm and did not consider any structural information derived from the specific documents.

For text mining tasks, the majority of state-of-the-art frameworks employ the vector space model (VSM), which treats a document as a bag of words and uses plain language words as features [13], [14]. This model can represent the text mining problems easily and directly. However, with the increase of data set size, the vector space becomes high-dimensional, sparse, and the computational complexity grows exponentially. Moreover, in many practical applications, completely unsupervised learning is lacking relevant information. On the other hand, supervised learning needs an initial large number of class label information, which requires expensive human labor and time [15], [16]. Therefore, in recent years, semisupervised learning has captured a great deal of attentions [17], [18], [19], [20], [21]. Semisupervised learning is a machine learning paradigm in which the model is constructed using both labeled and unlabeled data for training—typically a small amount of labeled data and a large amount of unlabeled data [16], [22].

- *R. Guan, X. Shi, and Y. Liang are with the College of Computer Science and Technology, Jilin University, No. 68 Mailbox, 2699 Qianjin Road, Changchun 130012, China. E-mail: guan_renchu@yahoo.com.cn, {shixh, ycliang}@jlu.edu.cn.*
- *M. Marchese is with the Department of Information Engineering and Computer Science, Faculty of Science, University of Trento, Via Sommarive, 14 I-38123, Trento 38100, Italy. E-mail: maurizio.marchese@unitn.it.*
- *C. Yang is with the College of Earth Sciences, Jilin University, 2199 Jianshe Road, Changchun 130061, China. E-mail: yangc_616@yahoo.com.cn.*

Manuscript received 14 Feb. 2009; revised 20 Oct. 2009; accepted 10 Dec. 2009; published online 24 Aug. 2010.
Recommended for acceptance by M. Ester.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2009-02-0070.
Digital Object Identifier no. 10.1109/TKDE.2010.144.

In this paper, we present a new clustering algorithm by extending Affinity Propagation with 1) a novel asymmetric similarity measurement that captures the structural information of texts, and 2) a semisupervised learning approach, where we exploit the knowledge from a few labeled objects versus a large number of unlabeled ones.

In information retrieval, there are several commonly used measurements of similarities. The simplest of all similarity measures, namely, simple matching coefficient, is counting the number of shared terms in two sets (e.g., documents)

$$S(X,Y) = |X \cap Y|, \tag{1}$$

where |Z| gives the size of a set Z. This coefficient does not take into account the sizes of X and Y. A more powerful coefficient called Cosine coefficient takes this information into account [23]:

$$S(X,Y) = \frac{|X \cap Y|}{|X|^{1/2}\,|Y|^{1/2}}. \tag{2}$$

Both metrics are symmetric. However, such symmetric property is not always present in realistic cases. For instance, to measure the similarity among sentences, Frey and Dueck [7] proposed to compute the similarity between sentence i and sentence k based on the cost of encoding the words in sentence i using the words in sentence k. This ad hoc similarity measurement performed very well. In their study, it is found that about 97 percent of such similarities were not symmetric.

Similarly, in this paper, we propose an asymmetric similarity measurement for two different documents, which is different from the conventional symmetric measurements [23], [24]. Specifically, in text clustering, text documents typically contain a large number of structural information which is completely ignored in conventional Cosine coefficient measures. For instance, in structured documents such as news articles and/or scientific papers, relevant words, or phrases emerge in specific sections of the document, i.e., title, author, abstract, keywords, and references (in scientific literatures) which may be considered as terms carrying structural information. Embracing some ideas of positive and negative association rules proposed in [25], we define three feature sets containing structural information. An asymmetric similarity measurement—called Tri-Set method—is thus proposed based on these three feature sets.

Finally, we present and analyze the definition of specific initial values for the clustering algorithm, that we named *Seeds*, to bootstrap the initial phases of the new clustering algorithm. We thus propose a novel semisupervised clustering algorithm: Seeds Affinity Propagation (SAP). This model aims to address the complexity problem in text clustering which results from the high dimension and sparse matrix computations.

To examine the effectiveness of the proposed method, we have applied it to the benchmark data set Reuters-21578. In order to analyze the behavior of the new algorithm (and also the impact of the two individual proposed contributions), we have performed a detail comparison with four clustering methods on the same data set, namely,

1. k-means approach;
2. the original Affinity Propagation algorithm with conventional similarity measurement (AP(CC));
3. a modified Affinity Propagation method, which combines AP with the new similarity measurement (AP(Tri-Set)); and
4. a modified Affinity Propagation method which combines AP with the new seed construction semisupervised method (SAP(CC)).

In our experiments, k-means (proposed by MacQueen [24] in 1967) is selected as the baseline state-of-the-art clustering algorithm. It is an important and successful method in data mining and knowledge discovering. Many algorithms are derived from k-means or compete with it (see, for instance, [26], [27], [28]). In particular, Wu et al. discussed several limitations about k-means in detail, such as its sensitivity to initialization, to the presence of outliers [29]. Our experimental results show that SAP offers better speed (i.e., about 20 times faster than k-means) and overall precision than the other four clustering algorithms (i.e., F-measures increase up to 44 percent compared with k-means).

The rest of the article is organized as follows: we start in Section 2 with a brief review of the Affinity Propagation clustering approach and related work. In Section 3, the definitions of structural information sets are introduced, and then, the SAP algorithm is described in detail. Section 4 presents the experimental methodology and results on a benchmark data set as well as the comparison with the selected baseline algorithms. Section 5 discusses conclusions and future work.

2 Related Work

AP was proposed as a new and powerful technique for exemplar learning. In brief, the user has to provide as initial input to the algorithm a complete matrix of similarities (for the selected metric(s)) among the input data points. At first, all data points are viewed as potential exemplars. Then, after a large number of real-valued information messages (i.e., named responsibility and availability messages, see below) are transmitted along the edges of the network (each data point is viewed as a node), a relevant set of exemplars and corresponding clusters is identified [7].

In the following, we detail in brief the mathematical model of the AP approach. At start-up, AP takes in input a collection of real-valued similarities between data points. Given an N data point's data set, x_i and x_j are two objects in it. The similarity $s(i, j)$ indicates how well x_j is suited to be the exemplar for x_i. For instance, it can be initialized to $s(i,j) = -\|x_i - x_j\|^2, i \neq j$. Hereinto, if there is no heuristic knowledge, self-similarities are called as *preference* in [7] and often set as a constant. For instance, they could be set as

$$s(l,l) = \frac{\sum_{i,j=1,i\neq j}^{N} s(i,j)}{N \times (N-1)} \quad 1 \leq l \leq N. \tag{3}$$

Then, the AP approach computes two kinds of messages exchanged between data points. The first one is called "*responsibility*" $r(i, j)$: it is sent from data point i to candidate exemplar point j and it reflects the accumulated

evidence for how well-suited point j is to serve as the exemplar for point i.

The second message is called "*availability*" $a(i, j)$: it is sent from candidate exemplar point j to point i and it reflects the accumulated evidence for how appropriate it would be for point i to choose point j as its exemplar. At the beginning, the availabilities are initialized to zero: $a(i, j)= 0$. The update equations for r (i, j) and $a(i, j)$ are written as

$$r(i,j) = s(i,j) - \max_{j' \neq j}\{a(i,j') + s(i,j')\}, \tag{4}$$

$$a(i,j) = \begin{Bmatrix} \min\left\{0, r(j,j) + \sum_{i' \neq i,j} \max\{0, r(i',j)\}\right\}, & i \neq j \\ \sum_{i' \neq i} \max\{0, r(i',j)\}, & i = j \end{Bmatrix}. \tag{5}$$

In addition, during each messages' exchange between data points, a damping factor $\lambda \in [0, 1]$ is added to avoid numerical oscillations that may arise in some circumstances:

$$\begin{aligned} R_{t+1} &= (1-\lambda)R_t + \lambda R_{t-1}, \\ A_{t+1} &= (1-\lambda)A_t + \lambda A_{t-1}, \end{aligned} \tag{6}$$

where $R = (r(i, j))$ and $A = (a(i, j))$ represent the responsibility matrix and availability matrix, respectively, and t indicates the iteration times.

The above two messages are updated iteratively, until they reach some specified values or the local decisions stay constant for a number of iterations. At this stage, availabilities and responsibilities can then be combined to identify exemplars:

$$c_j \leftarrow \arg\max_{1 \leq j \leq N}[r(i,j) + a(i,j)]. \tag{7}$$

Many detailed analyses of the AP approach have been carried out (see, for instance, [30] and [31]) for various data sets with different scales. These studies show that for small data sets, there are only minor differences between traditional strategies (such as p-median model and vertex substitution heuristic) and Affinity Propagation clustering for both precision and CPU execution time. Nevertheless, for large data sets, AP offers obvious advantages over existing methods [7], [31]. In particular, in their work, Frey and Dueck showed that an improvement in execution time of roughly 100 times is achieved on data sets of more than 10,000 objects and ca. 500 clusters. Moreover, in [7], [8], [9], [10], [11], [12], [30], [31], [32], [33], it has been identified that the similarity measurement has a great influence on AP clustering.

3 Seeds Affinity Propagation

Based on AP method, we propose a novel method called "Seeds Affinity Propagation." The main new features of the new algorithm are: *Tri-Set computation, similarity computation, seeds construction, and messages transmission*.

We start the presentation of the algorithm by explaining the basic similarity measurement used in our approach, i.e., three new feature sets, named by Cofeature Set (CFS), Unilateral Feature Set (UFS), and Significant Cofeature Set (SCS). The structural information of the text documents is included into the new similarity measurement. Then, we present how we extend the original AP approach with semisupervised learning strategy. The whole process of SAP is listed in Section 3.3.

3.1 Similarity Measurement

As discussed in Section 2, similarity measurement plays an important role in Affinity Propagation clustering. In order to give specific and effective similarity measurement for our particular domain, i.e., text document, we introduce the following feature sets: the Cofeature Set, the Unilateral Feature Set, and the Significant Cofeature Set. To define these sets, we first detail the computations of the new features. In our approach, each term in text is still deemed as a feature and each document is still deemed as a vector [23]. However, all the features and vectors are not computed simultaneously, but one at a time.

Let $D = \{d_1, d_2, \ldots, d_N\}$ be a set of texts. Suppose that d_i and d_j are two objects in D, they can be represented using the following two subsets:

$$\begin{aligned} d_i &= \{\langle f_i^1, n_i^1\rangle, \langle f_i^2, n_i^2\rangle, \ldots \langle f_i^L, n_i^L\rangle\}, \\ d_j &= \{\langle f_j^1, n_j^1\rangle, \langle f_j^2, n_j^2\rangle, \ldots \langle f_j^M, n_j^M\rangle\}, \end{aligned}$$

where f_i^x and f_j^y $(1 \leq x \leq L, 1 \leq y \leq M)$ in the two-tuples $\langle f_i^x, n_i^x\rangle, \langle f_j^y, n_j^y\rangle$ represent the xth and yth feature of d_i and d_j, respectively. (In most cases, when $x = y$, $f_i^x \neq f_j^y$.) n_i^x and n_j^y are the values of f_i^x and f_j^y. L and M are the counts of the objects' features.

Let F_i and F_j be the feature sets of the two objects, respectively: $F_i = \{f_i^1, f_i^2 \cdots f_i^L\}, F_j = \{f_j^1, f_j^2 \cdots f_j^M\}$. Let us introduce now, the set DF_j composed of the "most significant" features of d_j. "Most significant" means features that are capable of representing crucial aspects of the document. These "most significant" features could be key phrases and/or tags associated with each document when available. Or, as we have used in our experiments, they could be all the words (except stop words) in the title of each document.

Let $F_{(i,j)} = F_i \cap F_j,\ \bar{F}_{(i,j)} = F_i - F_{(i,j)},\ \hat{F}_{(i,j)} = F_{(i,j)} \cap DF_j$. The Venn diagrams related to these sets are showed in Fig. 1, where the hatching parts are $F_{(i,j)}, \bar{F}_{(i,j)}$, and $\hat{F}_{(i,j)}$. Consequently, it is easier to think of $F_{(i,j)} \cup \bar{F}_{(i,j)} = F_i,\ \hat{F}_{(i,j)} \subseteq F_{(i,j)}$. In special case: $\bar{F}_{(i,j)} = F_i,\ \hat{F}_{(i,j)} = \phi$, where $F_{(i,j)} = \phi$.

From the above formal objects, we can now define the three subsets in the two objects' vector space model.

Definition 1. *Cofeature Set.*

Let d_i and d_j be two objects in a data set. Suppose that some features of d_i, also belong to d_j. Consequently, we can construct a new two-tuples subset consisting of these features and their values in d_j.

We define it as the Cofeature Set between d_i and $dj : \langle f_m, n_m\rangle \in CFS_{(i,j)}$, iff $f_m \in F_{(i,j)}$, and $\langle f_m, n_m\rangle \in d_j$.

Definition 2. *Unilateral Feature Set.*

Suppose that some features of d_i, do not belong to d_j. Consequently, we can construct a new two-tuples subset consisting of these features and their values in d_i.

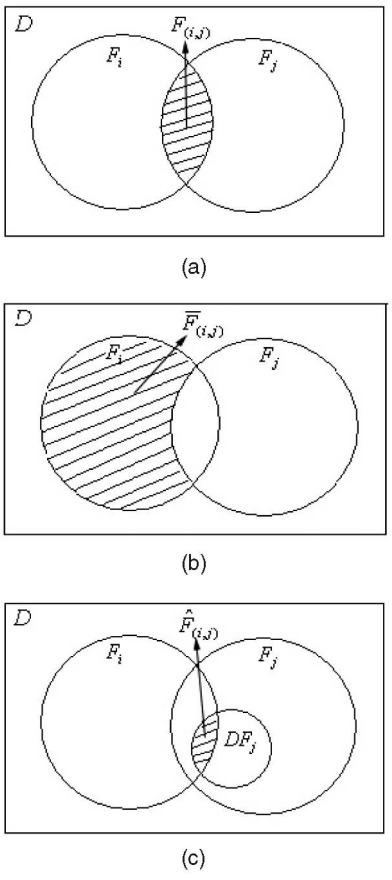

Fig. 1. Three kinds of relations between the two feature subsets of d_i and d_j. F_i and F_j are their feature subsets. DF_j is the most significant feature of d_j. D is the whole data set. (a) Venn diagram of $F_{(i,j)}$. (b) Venn diagram of $\bar{F}_{(i,j)}$. (c) Venn diagram of $\hat{F}_{(i,j)}$.

We define it as the Unilateral Feature Set between d_i and d_j: $\langle f_p, n_p\rangle \in UFS_{(i,j)}$ and $\langle f_m, n_m\rangle \in d_j$.

Definition 3. *Significant Cofeature Set.*

Suppose that some features of d_i, also belong to the most significant features of d_j. Consequently, we can construct a new two-tuples subset consisting of these features and their values as the most significant features in d_j.

We define it as the Significant Cofeature Set between d_i and d_j: $<f_q, n_q> \in SCS_{(i,j)}$, where $f_q \in \hat{F}_{(i,j)}$, n_q represent the value of feature f_q in DF_j.

In our proposed approach, we thus extend the generic definition of the similarity measures based on the Cosine coefficient by introducing the three new sets $CFS_{(i,j)}$, $UFS_{(i,j)}$, and $SCS_{(i,j)}$, namely, Tri-Set similarity:

$$s(i,j) = \alpha \sum_{m=1}^{|CFS|} n_m + \beta \sum_{q=1}^{|SCS|} n_q - \gamma \sum_{p=1}^{|UFS|} n_p, \qquad (8)$$

where CFS, UFS, and SCS, respectively, indicate the number of two-tuples in $CFS_{(i,j)}$, $UFS_{(i,j)}$, and $SCS_{(i,j)}$.

$CFS_{(i,j)}$ (i.e., $\sum_{m=1}^{|CFS|} n_m$) is derived from the intersection of two objects $|X \cap Y|$ as defined in the equation for the Cosine coefficient (see (2));

$UFS_{(i,j)}$ (i.e., $\sum_{p=1}^{|UFS|} n_p$) takes into account the unshared features; $SCS_{(i,j)}$ (i.e., $\sum_{q=1}^{|SCS|} n_q$) takes into account the most significant features.

The parameters α, β, and γ are adaptive factors of which the selection will be outlined in Section 4.

We believe that this extended similarity measure can reveal both the difference and the asymmetric nature of similarities between documents. Moreover, we think that it is more effective in the application of Affinity Propagation clustering for text documents, image processing, gene detecting, and so on, since it is capable to deal with asymmetric problems. We named the combination of this new similarity with conventional Affinity Propagation the Tri-Set Affinity Propagation (AP(Tri-Set)) clustering algorithm.

3.2 Seeds Construction

In semisupervised clustering, the main goal is to efficiently cluster a large number of unlabeled objects starting from a relatively small number of initial labeled objects. Given a few initial labeled objects, we would like to use them to construct efficient initial "seeds" for our Affinity Propagation clustering algorithm.

To guarantee precision and avoid a blind search for seeds and imbalance errors, we present in the following a specific seeds' construction method, that we named Mean Features Selection. Let N^O, N^F, N^D, and F_C represent, respectively, the object number, the feature number, the most significant feature number, and the feature set of cluster c in the labeled set (they can be searched by viewing each object in cluster c). Suppose F is the feature set and DF is the most significant feature set of seed c (for example, DF of this manuscript could be all the words (except stop words) in the title, i.e., {text, clustering, seed, Affinity, and Propagation}). Let $f_k \in F_C, f_{k'} \in F_C$. Their values in cluster c are n_k and nk', the values of being the most significant feature are $n_{DK}(0 \le n_{DK} \le n_k)$ and $n_{DK'}(0 \le n_{DK'} \le n_{K'})$.

The seeds' construction method is prescribed as

1. *iff*
$$n_{k'} \ge \frac{\sum_{k=1}^{N^F} n_k}{N^o}, \quad f_{k'} \in F;$$
2. *iff*
$$n_{DK'} \ge \frac{\sum_{k=1}^{N^D} n_{DK}}{N^o}, \quad f_{k'} \in DF.$$

This method can quickly find out the representative features in labeled objects. The seeds are made up of these features and their values in different clusters. Accordingly, they should be more representative and discriminative than normal objects.

In addition, for seeds, their self-similarities are set to $+\infty$ to ensure that the seeds will be chosen as exemplars and help the algorithm to get the exact cluster number.

We named the combination of this semisupervised strategy with classical similarity measurement and conventional Affinity Propagation as Seeds Affinity Propagation with Cosine coefficient (SAP(CC)) clustering algorithm. By

introducing both the seed construction method and the new similarity measurement into conventional AP, we arrive at the definition of the complete "Seeds Affinity Propagation algorithm," which will be detailed in the next section.

3.3 Seeds Affinity Propagation Algorithm

Based on the definitions of UFS, SCS, and the described seeds' construction method, the SAP algorithm is developed, following this sequence of steps:

1. *Initialization*: Let the data set D be an $N(N > 0)$ terms superset where each term consists of a sequence of two-tuples:
$$D = \left\{ \begin{array}{l} \{\langle f_1^1, n_1^1\rangle, \langle f_1^2, n_1^2\rangle, \ldots \langle f_1^{M^1}, n_1^{M^1}\rangle\}, \cdots \\ \{\langle f_N^1, n_N^1\rangle, \langle f_N^2, n_N^2\rangle, \ldots \langle f_N^{M^N}, n_N^{M^N}\rangle\} \end{array} \right\},$$
where M^x represents the count of the xth object's features.
2. *Seeds construction*: Constructing seeds from a few labeled objects according to Mean Features Selection. At this point, we have to define pragmatically the set *DF* (for instance, we can use the words in the title and/or key phrases or tags explicitly provided by the authors for text documents or the central part of an image in image clustering; in this paper, we used all the words—excluding stop words—in the title). Add these new objects into the data set D, and get a new data set D' which contains N' terms ($N \leq N'$).
3. *Tri-Set computation*: Computing the $(CFS_{(i,j)})$, $(UFS_{(i,j)})$, and $(SCS_{(i,j)})$ between objects i and j by using their definitions, where $i \in D'$ and $j \in D'$.
4. *Similarity computation*: Computing the similarities among objects in D' using (8).
5. *Self-Similarity computation*: Computing the self-similarities for each object in D'. For $i \in D'$, $j \in D'$, and $i \neq j$,φ is a preference value which can be seen as an adaptive factor. When x is a seed, its self-similarity is set to $+\infty$:
$$s(l,l) = \left\{ \begin{array}{ll} \varphi \dfrac{\sum_{i,j=1;i\neq j}^{N'} s(i,j)}{N' \times (N'-1)}, & 0 < l \leq N \\ +\infty, & N < l \leq N' \end{array} \right\}. \quad (9)$$
6. *Initialize messages*: Initializing the matrixes of messages
$$r(i,j) = s(i,j) - \max_{j \neq j'} \{s(i,j')\}, a(i,j) = 0, \quad (10)$$
where $i \in D', j \in D'$, and $i \neq j$.
7. *Message matrix computation*: Computing the matrixes of messages using (4) and (5).
8. *Exemplar selection*: Adding the two message matrixes and searching the exemplar for each object i which is the maximum of $r(i, j) + a(i, j)$.
9. *Updating the messages* using (6).
10. *Iterating steps 6, 7, and 8* until the exemplar selection outcome stays constant for a number of iterations or after a fixed number of iterations. End the algorithm.

To summarize, we start with the definition of three new relations between objects. Then, we assign the three feature sets with different weights and present a new similarity measurement. Finally, we define a fast initial seeds construction method and detail the steps of the new Seeds Affinity Propagation algorithm in the general case.

4 Experiments and Discussion

To examine the behavior and the performance of SAP algorithm, we have experimented on a widely used benchmark text data Reuters-21578 (Reuters) [34], [35], [36], [37], [38]. In order to compare the proposed SAP algorithm, we have performed the same clustering operation with two state-of-the-art clustering algorithms, namely, 1) k-means and 2) the original Affinity Propagation. Moreover, to further investigate the impact of the individual newly proposed contributions, we have also run Affinity Propagation algorithm using only the new Tri-Set Similarity metric (AP (Tri-Set)) and only seed construction semisupervised approach with the original similarity measure (SAP (CC)). For the comparison of the obtained results, we have adopted three standard quality measurement parameters, namely, F-measure, entropy, and CPU execution time. In addition, we have also investigated the robustness of the proposed algorithm on different data distribution. Since previous work on the original AP (and subsequent improvements) did not pay attention to this important parameter, we have examined and compared the robustness for all the five algorithms.

4.1 Experimental Setup

The publicly available Reuters-21578 (Reuters) data set is preclassified manually [36]. This classification information is eliminated before the clustering processes, and is used to evaluate the clustering accuracy of each clustering algorithm at the end of the execution. The original Reuters data consist of 22 files (for a total of 21,578 documents) and contain special tags such as "<TOPICS>" and "<DATE>" among others. The preprocessing phase on the data set cuts the files into single texts and strips the document from the special tags. Then, those documents which belong to at least one topic are selected. At last, after stop words removal, word stemming, and word frequency computation for each document, the data set turns into the form of
$$D = \left\{ \begin{array}{l} \{\langle f_1^1, n_1^1\rangle, \langle f_1^2, n_1^2\rangle, \ldots \langle f_1^{M^1}, n_1^{M^1}\rangle\}, \ldots \\ \{\langle f_N^1, n_N^1\rangle, \langle f_N^2, n_N^2\rangle, \ldots \langle f_N^{M^N}, n_N^{M^N}\rangle\} \end{array} \right\}.$$

For text clustering problem, Cofeature Set can be viewed as a two-tuples set. Each term in the set consists of one word that exists both in d_i and d_j, and the word's frequency in d_j. The terms in the Unilateral Feature Set, on the other hand, consist of the words that only exist in d_i and their frequencies in d_i. Moreover, there are some words that exist in the title, abstract, or in the first sentence of each paragraph in d_j and they can also be found in d_i. These words and their frequencies at important positions of d_j can be viewed as the two-tuples of the Significant Cofeature Set (we used the words (except stop words) in the title). For the construction of seeds, in order to quickly find out the representative features, the proposed Mean Features Selection strategy is applied.

4.2 Evaluation Measures

To evaluate the performance of clustering, three kinds of measures, F-measure, entropy, and CPU execution time, are used to compare the generated clusters with the set of categories created manually in Reuters. The F-measure is a harmonic combination of the precision and recall values used in information retrieval. Therefore, we first calculate the precision $P(i, j)$ and recall $R(i, j)$ of each cluster j for each class i, based on the results of the clustering algorithms and on the classification information provided with the Reuters-21578 (Reuters) data set. Precision and Recall can be defined as

$$P(i,j) = \frac{N_{ij}}{N_j}, \tag{11a}$$

$$R(i,j) = \frac{N_{ij}}{N_i}, \tag{11b}$$

where N_{ij} is the number of objects of class i in cluster j, N_j is the number of objects of cluster j, and N_i is the number of objects of class i. The corresponding F-measure $F(i, j)$ is defined as

$$F(i,j) = \frac{2P(i,j)R(i,j)}{P(i,j) + R(i,j)}. \tag{12}$$

The global F-measure for the whole clustering result is defined as

$$F = \sum_i \frac{N_i}{N} \max_j (F(i,j)), \tag{13}$$

where N is the total number of documents in the data set. Due to the higher accuracy of the clusters mapping to the original classes, the larger the F-measure, the better the clustering performance.

Entropy provides a measure of the uniformity or purity of a cluster. In other words, it can tell us how homogeneous a cluster is. The smaller the entropy, the better the clustering performance. The entropy of each cluster j is calculated using the standard equation

$$E_j = -\sum_i p_{ij} \log(p_{ij}), \tag{14}$$

where p_{ij} is the probability of an object that belongs to class i in cluster j. Then, the total entropy for a set of clusters is calculated as the sum of entropies of each cluster:

$$E = \sum_{j=1}^{m} \left(\frac{N_j}{N} \times E_j \right), \tag{15}$$

where N is the total number of objects in the data set, N_j is the number of objects of cluster j, and m is the number of clusters.

The last metric—the CPU execution time—provides us a measure of the efficiency and scalability of the algorithm when large data set is used.

4.3 General Comparison

The experiments use the top 10 classes ("acq," "corn," "crude," "earn," "grain," "interest," "money-fx," "ship," "trade," and "wheat") extracted from Reuters, which have been widely used in the information retrieval area [37, 38]. To examine the efficiency of our approach to different collection size with each topic on discrete uniform distribution, all approaches are applied to data sets with 400, 600, 800, 1,000, 1,200, and 1,400 texts, respectively. All the experiments run on a PC (Intel (R) Pentium (R) D CPU 2.8 GHz, 2.8 GHz with 1 GB of RAM).

TABLE 1
Different Strategies for K-Means, AP (CC), AP (TRI-SET), SAP (CC), and SAP

	k-means	AP(CC)	AP(Tri-Set)	SAP(CC)	SAP
Tri-Set Similarity	×	×	√	×	√
Semi-Supervision	×	×	×	√	√

We have applied five different clustering algorithms on these data sets. In Table 1, we list both the algorithms and the use of the new contributions: k-means, AP (CC), and SAP (CC) use Cosine coefficient as similarity measurement, while AP (Tri-Set) and SAP utilize Tri-Set Similarity measure; SAP (CC) and SAP apply semisupervision strategy, while all other algorithms do not use such strategy. Some details of the five algorithms' strategies are described in the following:

K-means algorithm adopts the widely used similarity measurement Cosine coefficient, namely, (16). Because k-means needs to compute on the vector space model, for $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_n)$, the component form of their similarity measurement can be expressed as

$$S(X,Y) = \frac{\sum_{i=1}^{n} x_i y_i}{\left(\sum_{i=1}^{n} x_i^2\right)^{1/2} \left(\sum_{i=1}^{n} y_i^2\right)^{1/2}}, \tag{16}$$

where n is the number of the features in the whole vector space [23].

Similar to k-means, AP (CC) and SAP (CC) also use Cosine coefficient ((2)) as the similarity between two documents. However, unlike k-means, a document in AP (CC) and SAP (CC) does not need to be represented into the whole vector space, but only into its own vector space. Therefore, the similarity measurement computation complexity of the latter two algorithms is reduced greatly in respect to one of the k-means. The self-similarities of AP (CC) are defined as [39]

$$\begin{aligned} s(l,l) = & \min_{1 \le i,j \le N, i \ne j} \{s(i,j)\} \\ & - \varphi \left(\max_{1 \le i,j \le N, i \ne j} \{s(i,j)\} - \min_{1 \le i,j \le N, i \ne j} \{s(i,j)\} \right). \end{aligned} \tag{17}$$

The self-similarities of SAP (CC), with semisupervised strategy, are computed as

$$s(l,l) = \begin{cases} \min\limits_{1 \le i,j \le N', i \ne j} \{s(i,j)\} \\ -\varphi \left(\max\limits_{1 \le i,j \le N', i \ne j} \{s(i,j)\} - \min\limits_{1 \le i,j \le N', i \ne j} \{s(i,j)\} \right) & 1 < l \le N, \\ +\infty, & N < l \le N', \end{cases} \tag{18}$$

where φ is an adjustable factor.

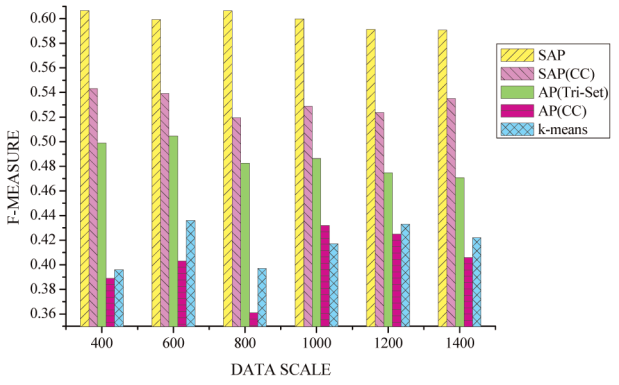

Fig. 2. F-measure comparison.

For AP (Tri-Set) and SAP, the similarities between two different texts are measured by Tri-Set method using (8). However, the self-similarities of AP (Tri-Set) are set as a constant:

$$s(l,l) = \frac{\sum_{i,j=1;i\neq j}^{N} s(i,j)}{N \times (N-1)} - \varphi \max_{1\leq i,j\leq N}\{S(i,j)\}, \qquad (19)$$

where φ is an adjustive factor. The self-similarities of SAP, with semisupervised strategy, are computed using (9).

For semisupervised learning strategy of SAP (CC) and SAP, four labeled documents are merged for a seed using the Mean Features Selection method and each class owns one seed.

Figs. 2, 3, and 4 show the comparisons for F-measure, Entropy, and CPU execution time for the five algorithms, respectively. In Fig. 2 and the summary results in Table 2, it can be seen that the average F-measure value of AP (CC) is close to that of k-means, while the average F-measures of AP (Tri-Set), SAP (CC), and SAP are 16.8, 27.6, and 43.9 percent higher than that of k-means, respectively.

Fig. 3 and Table 2 show an extremely different trend of the Entropy values for the five methods. Those of k-means and AP (CC) are close and they show the highest entropies; AP (Tri-Set) entropy is about 7.3 percent lower than that of k-means, on average; SAP (CC) is about 10.0 percent lower than k-means; the lowest one is SAP: it is 28.3 percent lower than that of k-means, on average, and 28.2 percent lower than the original AP(CC) algorithm.

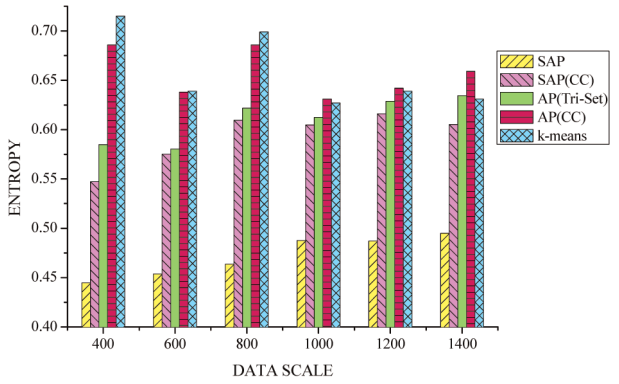

Fig. 3. Entropy comparison.

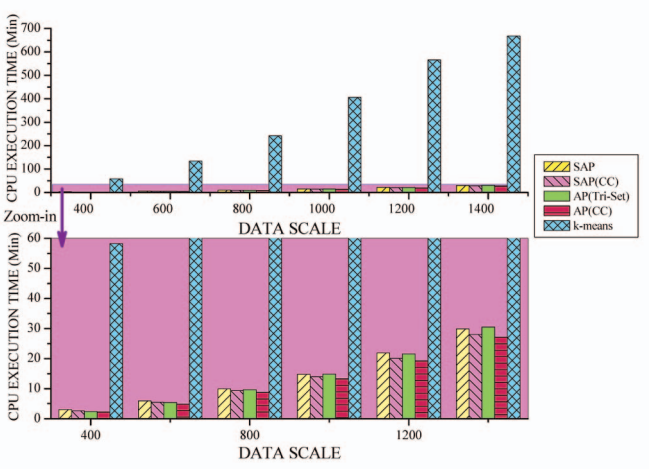

Fig. 4. CPU execution time comparison.

From Fig. 4, it is clear that the CPU execution time of all Affinity Propagation-based algorithms—SAP, SAP (CC), AP (Tri-Set), and AP (CC)—is far less than that of k-means, and the gaps enlarge exponentially when the data set scale increases. For example, k-means consumes about 18.1 times larger than SAP for a 400-document data set, while this number is increased to 21.3 for a 1,400-document data set. The most significant advantage of SAP is that it is better than k-means in the foregoing evaluations, while k-means runs 200 times (the best run is used to compare with SAP) and costs about 20-fold of SAP in time. This result also confirms the one in [7], [30], and [31]. Furthermore, even after 10,000 runs of k-means—with a size of 400 documents (F-measure: 0.406; Entropy: 0.677), we can't get similar results as SAP. To get the totally best result of k-means, it needs to execute all possible solutions. That is, at least $C_{400}^{10} \approx 2.57 \times 10^{19}$ k-means runs need to be performed for 400 documents.

In addition, the zoom-in figure of Fig. 4 and the summary results in Table 2 show that AP (CC) is about 10 percent faster than AP (Tri-Set). Similarly, SAP (CC) uses about 7 percent less time than SAP. This is due to the fact that Cosine coefficient computing is generally faster. However, Tri-Set similarity performs much better than Cosine coefficient on F-measure and Entropy. A good case in point is the comparison of AP (Tri-Set) and AP (CC) in Figs. 2 and 3. On average of F-measure, the former is 20.6 percent higher. On average of Entropy, AP (Tri-Set) is 7.1 percent lower than AP (CC).

TABLE 2
Mean Values over All Experiments

	Mean F-MEASURE	Mean Entropy	Mean CPU execution time
SAP	**0.599**	**0.472**	14.3
SAP (CC)	0.531	0.592	13.3
AP(Tri-Set)	0.486	0.610	14.1
AP(CC)	0.403	0.657	**12.6**
K-means	0.416	0.658	345.9

TABLE 3
List of AP (CC), AP (TRI-SET), SAP (CC), and SAP Adjustive Factors

	AP (CC)	AP (Tri-Set)			SAP(CC)	SAP			
	φ	α	β	γ	φ	φ	α	β	γ	φ
400	2.59	1	13	2	2	4	1	10	3	6
600	3.40	1	13	4	4	6	1	9	3	8
800	4.50	1	11	3	6	7	1	10	4	10
1000	5.48	1	15	2	5	10	1	12	4	14
1200	6.20	1	19	3	4	11	1	13	4	18
1400	7.50	1	20	4	7	13	1	14	4	19

TABLE 4
General Comparison Experiment Parameters

	Documents Number of Each Class	Number of Words	Average Words Number of each document	Seeds percentage
400	40	5585	60.85	10.0%
600	60	6884	59.83	6.7%
800	80	7930	59.81	5.0%
1000	100	8838	59.05	4.0%
1200	120	9584	59.37	3.3%
1400	140	10316	60.46	2.9%

The adjustable factors of the four AP-based algorithms are listed in Table 3. For the three adaptive factors related to the Cofeature set, Unilateral Feature Set, and Significant Cofeature Set (used in AP(Tri-Set) and SAP), namely, α, β, and γ at the beginning, we sampled different values in order to obtain optimal clustering results as well as guidelines for their selection. After a large number of attempts, we found out that these three parameters are proportional (e.g., cluster results of $\alpha = 1$, $\beta = 1$, and $\gamma = 1$ are similar to $\alpha = 2$, $\beta = 2$, and $\gamma = 2$). Consequently, a simple way to proceed is to fix CFS parameter α to 1 and adjust the other two. We also observed that clustering results improve when β (i.e., the adaptive factor for SCS) is kept in the range {10-20} and is increased with data set size. For the last parameter γ (adaptive factor for UFS), we found optimal clustering results when it is chosen in the range {2, 3, 4, and 5}.

In all our experiments, the parameter φ (used in the self-similarity computation as an adaptive preference value) is correlated with data set size, following [39]. In fact, in [39], Frey et al. pointed out that the higher values of the preference parameter φ will cause AP to find more clusters, given the same amount of data. Following these empirical rules, we arrived to the definition of a set of adaptive parameters listed in Table 3.

Finally, the parameter k in k-means for all the experiments is set to 10.

From Table 4, it can be noted that how fast the computational complexity grows by increasing the data set size. For instance, though each document only has 60.46 words (on average) in the 1,400 data set, k-means method using (16) considers each document as a 10,316-dimension vector. Then, the problem is mapped into a large sparse matrix, and the time utilization is dramatically increased. On the contrary, SAP, SAP (CC), AP (Tri-Set), and AP (CC) need not to compute on the whole vector space. Therefore, AP-based algorithms are performed on a much smaller vector space than k-means.

To exam the effectiveness of semisupervised strategy, we plot the net similarities curves of AP (CC) and SAP (CC) in Fig. 5. It takes 400 texts as an example and the net similarity of iteration each has been calculated. Net similarity is the objective function that AP tries to maximize [7], [37]. SAP (CC) uses less iterations and earns high net similarities. When they converge, SAP (CC) is 11.67 percent higher than AP (CC) on net similarity and only uses 76.37 percent of AP (CC) iterations.

According to the discussion and figures above, it can safely draw the conclusion that SAP is superior to the other four algorithms. With the help of the new similarity measurement and the addition of the seeds, SAP greatly enhances the clustering performance. The cluster performance becomes more and more obvious when changes are step-up from AP (CC) to SAP: first of all, using Tri-Set Similarity instead of Cosine coefficient, the former obtains higher F-measure and lower entropy (see AP (CC) and AP (Tri-Set) comparison in Figs. 2 and 3). This is because the Tri-Set similarity contains the structural information which is omitted by CC; second, by introducing semisupervised strategy, the algorithms with seeds show better result than the ones without (see Fig. 5). This is because the semisupervised strategy can help to achieve a better solution and speed up convergence. In addition, for a similar CPU execution time, SAP with both two contributions obtains higher F-measure than AP (CC), AP (Tri-Set), and SAP (CC). Finally, with the growth of data set, SAP has a steady advantage on both F-measure and Entropy.

4.4 Robustness Comparison

Compared with discrete uniform distribution, nonuniform distribution of different categories is more familiar in the real world. Taking the Reuters as an example, the class

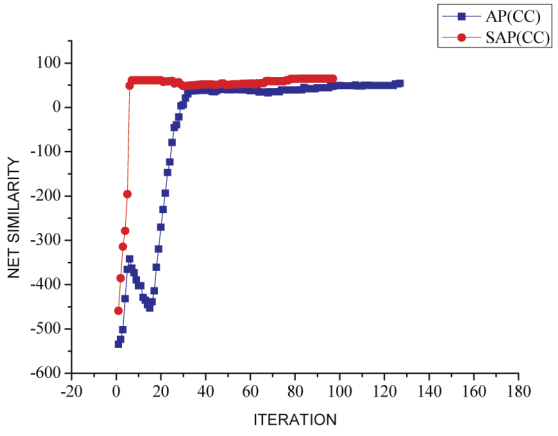

Fig. 5. Net similarities comparison.

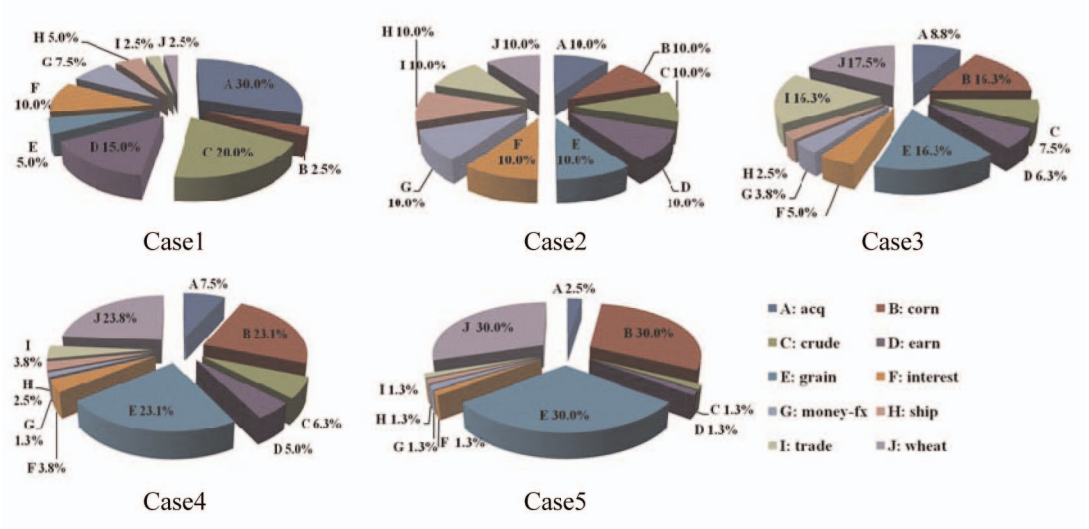

Fig. 6. Distribution of different topics: in case1, 10 percent "difficult;" in case2, 30 percent "difficult;" in case3, 50 percent "difficult;" in case4, 70 percent "difficult;" and in case5, 90 percent "difficult." In Reuters, corn, grain, and wheat documents are more similar to one another. The distances among these three classes are too small to identify. It means that they are too difficult to distinct. So, the distribution of these three classes can profoundly influence the clustering results. In Figs. 6a, 6b, 6c, 6d, and 6e, corn, grain, and wheat contain 10 percent (80 documents), 30 percent (240 documents), 50 percent (400 documents), 70 percent (560 documents), and 90 percent (720 documents), respectively.

distribution is very unbalanced. The set has 135 "TOPICS" categories, with the most frequent category ("earn") containing 3,987 documents and some small categories, such as "sun-meal," "castor-oil," and "lin-oil," only containing two documents. This inconsistency may affect the clustering results and it should be considered in evaluating the robustness of the cluster algorithms.

Based on this situation, an experiment with five cases is performed to examine the robustness of k-means, AP (CC), AP (Tri-Set), SAP (CC), and SAP. For each case, 800 texts which contain 10 classes are used. And each class has different number of documents. The distributions of different topics in the test are shown in Fig. 6.

From the nonuniform distribution experiment results in Tables 5, 6, and 7, it can also be noticed that the distribution of corn, grain, and wheat classes can profoundly influence the whole clustering results. With the growth of the percentage of the "difficult" classes, the precision of clustering algorithm is decreasing.

For all the three measurements and five different nonuniform distribution data sets (focus on the "difficult" classes percentage), SAP performs better than k-means, AP (CC), AP (Tri-Set), and SAP (CC) although AP (CC) is the fastest one. From Table 5, the average F-measure value of AP (CC) and k-means is much similar (k-means is 5.2 percent higher than AP (CC)). In the mean time, those of AP (Tri-Set), SAP (CC), and SAP are 16.3, 37.9, and 61.7 percent higher than that of k-means, respectively. Table 6 shows Entropy comparison of the five algorithms. AP (Tri-Set), SAP (CC), and SAP perform better than AP (CC) and k-means again. On average, AP (CC) is similar to k-means (AP (CC) is 0.6 percent lower than k-means); however, AP (Tri-Set), SAP (CC), and SAP are 5.9, 12.6, and 27.8 percent lower than k-means. From Table 7, we can see that k-means costs 20-folds time of AP (CC), AP (Tri-Set), SAP(CC), and SAP.

TABLE 5
F-measure Comparison with Nonuniform Distribution

	Case1	Case2	Case3	Case4	Case5	Mean
SAP	**0.749**	**0.606**	**0.573**	**0.544**	**0.489**	**0.592**
SAP (CC)	0.662	0.519	0.511	0.450	0.385	0.505
AP(Tri-Set)	0.577	0.482	0.419	0.364	0.290	0.426
AP(CC)	0.450	0.361	0.392	0.314	0.225	0.348
K-means	0.518	0.397	0.368	0.280	0.269	0.366

TABLE 6
Entropy Comparison with Nonuniform Distribution

	Case1	Case2	Case3	Case4	Case5	Mean
SAP	**0.325**	**0.463**	**0.484**	**0.492**	**0.487**	**0.450**
SAP (CC)	0.444	0.609	0.576	0.563	0.533	0.545
AP(Tri-Set)	0.486	0.621	0.626	0.609	0.594	0.587
AP(CC)	0.525	0.686	0.632	0.653	0.602	0.620
K-means	0.491	0.699	0.661	0.676	0.595	0.624

TABLE 7
CPU Execution Time Comparison with Nonuniform Distribution (Min)

	Case1	Case2	Case3	Case4	Case5	Mean
SAP	8.7	9.9	11.1	10.0	9.8	9.9
SAP (CC)	8.3	9.3	10.1	9.0	8.9	9.1
AP(Tri-Set)	8.7	9.7	10.6	9.4	9.2	9.5
AP(CC)	**7.8**	**8.7**	**9.6**	**8.4**	**8.7**	**8.7**
K-means	226.3	242.4	218.7	184.1	178.2	209.9

By analyzing the details of the clustering results, we could further conclude that SAP is more robust than the other four algorithms. Not only because it outperforms the other algorithms on evaluation measures, but also we found that SAP could catch the less represented topics. A good example is that in case 1, SAP works out a cluster, which is confirmed as "trade." This cluster contains 20 documents and 14 documents originally belong to "trade" in the Reuters classification. What's more is that the document number of this cluster is the same as the "trade" documents in the data set. On the contrary, k-means is trapped into putting all "trade" documents (20 documents) into a large cluster which includes all the 10 topics and 240 documents; AP (CC) is also trapped into putting most of "trade" documents (12 documents) into a large cluster which includes 10 topics and 173 documents; however, AP (Tri-Set) is better than the former two algorithms. It puts 16 "trade" documents into a smaller cluster which includes eight topics and 47 documents. SAP (CC) is similar to AP (Tri-Set). It also makes 16 "trade" documents into a smaller cluster; however, the cluster contains seven topics and 49 documents.

Based on all the experimental results above, an original analysis can be given: k-means is based on the objective function and searches for the minimum on coordinate descent [40]. Due to the nature of greedy-descent algorithm, the search is led to the direction of energy reduction [29], [41]. In this case, k-means is easy to be trapped into a local minimum in which it could get stuck in a suboptimal solution or may not converge when the data contain many classes with different sizes. However, SAP calculates similarity using Tri-Set method which considers different feature sets, adopts the max-sum algorithm, adds dumped factor, and introduces seeds that can definitely lead the algorithm to converge more quickly to the correct direction. Therefore, SAP can more efficiently work out the solution and avoid suboptimal solutions.

5 Conclusions

In this paper, we first proposed a similarity measurement which is extended from Cosine coefficient using structural information on the basis of Cofeature Set, Unilateral Feature Set, and Significant Cofeature Set. These three sets represent different features at different positions of texts. Their structural information improves the clustering results. The new similarity measurement can be used to calculate the asymmetric similarity directly, which is not limited to the symmetric space. Moreover, a new clustering algorithm which combines Affinity Propagation with semisupervised learning, namely, Seeds Affinity Propagation algorithm is proposed. SAP is applied to full text clustering which extends the application of Affinity Propagation. In the comparison with the classical clustering algorithm k-means, SAP not only reduces the computing complexity of text clustering and improves the accuracy, but also effectively avoids being random initialization and trapped in local minimum. SAP is also more robust and less sensitive to data distribution than k-means, conventional AP, AP (Tri-Set), and SAP (CC). In other words, it makes an important improvement in text clustering tasks. In addition, we believed that since SAP is based on a detailed similarity measurement and on a generic seeds construction strategy, it can be widely applied to other clustering problem domains. This is what we want to explore in our future work.

Acknowledgments

Professor Yanchun Liang was the corresponding author for this paper. This work was supported in part by the European Commission under grant No. 155776-EM-1-2009-1-IT-ERAMUNDUS-ECW-L12, the National Natural Science Foundation of China (60673023, 10872077, 60703025, 10501017), the Science-Technology Development Project from Jilin Province (20080708, 20090152), the National High-technology Development Project (2009AA02Z307), and the Graduate Innovation Fund Project (20091024) of Jilin University.

References

[1] Y.J. Li, C. Luo, and S.M. Chung, "Text Clustering with Feature Selection by Using Statistical Data," *IEEE Trans. Knowledge and Data Eng.,* vol. 20, no. 5, pp. 641-652, May 2008.

[2] C. Buckley and A.F. Lewit, "Optimizations of Inverted Vector Searches," *Proc. Ann. ACM SIGIR,* pp. 97-110, 1985.

[3] N. Jardin and C.J. van Rijsbergen, "The Use of Hierarchic Clustering in Information Retrieval," *Information Storage and Retrieval,* vol. 7, no. 5, pp. 217-240, 1971.

[4] G. Salton, *Dynamic Information and Library Processing.* Prentice-Hall, Inc., 1975.

[5] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval.* McGraw Hill Book Co., 1983.

[6] E.M. Voorhees, "The Efficiency of Inverted Index and Cluster Searches," *Proc. Ann. Int'l ACM SIGIR,* pp. 164-174, 1986.

[7] B.J. Frey and D. Dueck, "Clustering by Passing Messages between Data Points," *Science,* vol. 315, no. 5814, pp. 972-976, Feb. 2007.

[8] B.J. Frey and D. Dueck, "Non-Metric Affinity Propagation for Unsupervised Image Categorization," *Proc. 11th IEEE Int'l Conf. Computer Vision (ICCV '07),* pp. 1-8, Oct. 2007.

[9] L. Michele, Sumedha, and W. Martin, "Clustering by Soft-Constraint Affinity Propagation Applications to Gene-Expression Data," *Bioinformatics,* vol. 23, no. 20, pp. 2708-2715, Sept. 2007.

[10] T.Y. Jiang and A. Tuzhilin, "Dynamic Micro Targeting: Fitness-Based Approach to Predicting Individual Preferences," *Proc. Seventh IEEE Int'l Conf. Data Mining (ICDM '07),* pp. 173-182, Oct. 2007.

[11] H.F. Ma, X.H. Fan, and J. Chen, "An Incremental Chinese Text Classification Algorithm Based on Quick Clustering," *Proc. 2008 Int'l Symp. Information Processing (ISIP '08),* pp. 308-312, May 2008.

[12] W.H. Wang, H.W. Zhang, F. Wu, and Y.T. Zhuang, "Large Scale of E-Learning Resources Clustering with Parallel Affinity Propagation," *Proc. Int'l Conf. Hybrid Learning 2008 (ICHL '08),* pp.1-10, Aug. 2008.

[13] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys,* vol. 34, pp. 1-47, 2002.

[14] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, 1989.

[15] F. Wang and C.S. Zhang, "Label Propagation through Linear Neighbourhoods," *IEEE Trans. Knowledge and Data Eng.,* vol. 20, no. 1, pp. 55-67, Jan. 2008.

[16] Z.H. Zhou and M. Li, "Semi-Supervised Regression with Co-Training Style Algorithms," *IEEE Trans. Knowledge and Data Eng.,* vol. 19, no. 11, pp. 1479-1493, Aug. 2007.

[17] A. Blum and T. Mitchell, "Combining Labeled and Unlabeled Data with Co-Training," *Proc. 11th Ann. Conf. Computational Learning Theory,* pp. 92-100, 1998.

[18] S. Yu, B. Krishnapuram, R. Rosales, H. Steck, and R.B. Rao, "Bayesian Co-Training," *Advances in Neural Information Processing Systems,* vol. 20, pp. 1665-1672, MIT Press, 2008.

[19] M. Belkin and P. Niyogi, "Semi-Supervised Learning on Riemannian Manifolds," *Machine Learning,* vol. 56, pp. 209-239, 2004.
[20] Z.H. Zhou, D.C. Zhan, and Q. Yang, "Semi-Supervised Learning with Very Few Labeled Training Examples," *Proc. 22nd AAAI Conf. Artificial Intelligence,* pp. 675-680, 2007.
[21] Z.H. Zhou and M. Li, "Tri-Training: Exploiting Unlabeled Data Using Three Classifiers," *IEEE Trans. Knowledge and Data Eng.,* vol. 17, no. 11, pp. 1529-1541, Nov. 2005.
[22] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning.* MIT Press, 2006.
[23] C.J. van Rijsbergen, *Information Retrieval,* second ed., pp. 22-28. Butterworth, 1979.
[24] J. MacQUEEN, "Some Methods for Classification and Analysis of Multivariate Observations," *Proc. Fifth Berkeley Symp. Math. Statistics and Probability,* pp. 281-297, 1967.
[25] X.D. Wu and C.Q. Zhang, "Efficient Mining of Both Positive and Negative Association Rules," *ACM Trans. Information Systems,* vol. 22, no. 3, pp. 381-405, July 2004.
[26] M.J. Maña-López, M.D. Buenaga, and J.M. Gómez-Hidalgo, "Multidocument Summarization: An Added Value to Clustering in Interactive Retrieval," *ACM Trans. Information Systems,* vol. 22, no. 2, pp. 215-241, Apr. 2004.
[27] L.P. Jing, M.K. Ng, and J.Z. Huang, "An Entropy Weighting K-Means Algorithm for Subspace Clustering of High-Dimensional Sparse Data," *IEEE Trans. Knowledge and Data Eng.,* vol. 19, no. 8, pp. 1026-1041, Aug. 2007.
[28] S. Huang, Z. Chen, Y. Yu, and W.Y. Ma, "Multitype Features Coselection for Web Document Clustering," *IEEE Trans. Knowledge and Data Eng.,* vol. 18, no. 4, pp. 448-458, Apr. 2006.
[29] X.D. Wu et al., "Top 10 Algorithms in Data Mining," *Knowledge and Information Systems,* vol. 14, no. 1, pp. 1-37, Jan. 2008.
[30] M.J. Brusco and H.F. Kohn, "Comment on 'Clustering by Passing Messages between Data Points,'" *Science,* vol. 319, no. 5864, p. 726c, Feb. 2008.
[31] B.J. Frey and D. Dueck, "Response to Comment on 'Clustering by Passing Messages Between Data Points,'" *Science,* vol. 319, no. 5864, p. 726d, Feb. 2008.
[32] J. Wu, F. Ding, and Q.L. Xiang, "An Affinity Propagation Based Method for Vector Quantization," Eprint arXiv 0710.2037, http://arxiv.org/abs/0710.2037v2, Oct. 2007.
[33] K.J. Wang, J.Y. Zhang, D. Li, X.N. Zhang, and T. Guo, "Adaptive Affinity Propagation Clustering," *Acta Automatica Sinica,* vol. 33, no. 12, pp. 1242-1246, Dec. 2007.
[34] S. Gao, W. Wu, C.H. Lee, and T.S. Chua, "A Maximal Figure-of-Merit (MFoM)-Learning Approach to Robust Classifier Design for Text Categorization," *ACM Trans. Information Systems,* vol. 24, no. 2, pp. 190-218, Apr. 2006.
[35] Z.H. Zhou and M. Li, "Distributional Features for Text Categorization," *IEEE Trans. Knowledge and Data Eng.,* vol. 21, no. 3, pp. 428-442, Mar. 2009.
[36] D.D. Lewis, "Reuters-21578 Text Categorization Test Collection," http://www.daviddlewis.com/resources/testcollections/reuters21578, May 2004.
[37] A. Estabrooks, T. Jo, and N. Japkowicz, "A Multiple Resampling Method for Learning from Imbalanced Data Sets," *Computational Intelligence,* vol. 2, no. 1, pp. 18-36, 2004.
[38] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive Learning Algorithms and Representations for Text Categorization," *Proc. Seventh Int'l Conf. Information and Knowledge Management,* pp. 148-155, 1998.
[39] http://www.psi.toronto.edu/affinitypropagation/faq.html, 2010.
[40] H. Zha, C. Ding, M. Gu, X. He, and H.D. Simon, "Spectral Relaxation for K-Means Clustering," *Advances in Neural Information Processing Systems,* vol. 14, pp. 1057-1064, MIT Press, 2001.
[41] L. Bottou and Y. Bengio, "Convergence Properties of the K-Means," *Advances in Neural Information Processing Systems,* vol. 7, pp. 585-592, MIT Press, 1995.

Renchu Guan received the BSc degree in computer science and technology and the MSc degree in computer applied technology in 2004 and 2007 from Northeast Normal University, China, respectively. He received the PhD degree in computer science and technology (Bioinformatics) in 2010 from Jilin University, China, where he is currently an assistant professor in the College of Computer Science and Technology. He worked as an exchange student at the University of Trento, Italy, from June 2008 to December 2008. His research interests include machine learning, text mining, and bioinformatics.

Xiaohu Shi received the PhD degree in computer application technology in 2006 from Jilin University, Changchun, China, where he is currently an associate professor with the College of Computer Science and Technology. He has published more than 50 journal and conference papers. His current research interests include computing intelligent and bioinformatics.

Maurizio Marchese received the graduate degree in physics in 1984 from the University of Trento, Italy, where he is currently an associate professor of computer science in the Department of Information Engineering and Computer Science. He is the author of more than 80 publications and has been a program coordinator of a number of European Research Projects. His main research interests are the design and development of service architectures in distributed systems; the integration of services in Geographical Information Systems (GIS) environments; and the analysis, development, and integration of services to support and enhance the scientific knowledge creation and dissemination processes.

Chen Yang received the PhD degree in digital geosciences from Jilin University, China, where she is currently an assistant professor in the College of Earth Sciences. She worked as an exchange student at the University of Trento, Italy, from June 2008 to December 2008. Her research interests include remote sensing image processing, spatial data mining, and machine learning.

Yanchun Liang received the PhD degree in applied mathematics in 1997 from Jilin University, Changchun, China, where he is currently a professor in the College of Computer Science and Technology. He was a visiting scholar at Manchester University of United Kingdom from 1990 to 1991, a visiting professor at the National University of Singapore from 2000 to 2001, a guest professor in the Institute of High Performance Computing of Singapore from 2002 to 2004, and a guest professor at Trento University, Italy, from 2006 to 2008. He has published more than 280 papers. His research was featured in the *Bioinformatics*, *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, Journal of Micromechanics and Microengineering*, *Physical Review E*, *Neural Computing and Applications*, *Smart Materials and Structures*, *Artificial Intelligence in Medicine*, *Applied Artificial Intelligence*, etc. He was the recipient of several grants from NSFC, EU, etc. His research interests include computational intelligence, machine learning methods, text mining, MEMS modeling, and bioinformatics.