Procesamiento del Lenguaje Natural, núm. 43 (2009), pp. 245-253 recibido 1-05-2009; aceptado 5-06-2009

Fuzzy clusterers combination by positional voting for robust document clustering

Combinación de clusterizadores difusos mediante voto posicional para clustering robusto de documentos

Xavier Sevillano, Joan Claudi Socoró, Francesc Alías
GTM – Grup de Recerca en Tecnologies Mèdia
La Salle. Universitat Ramon Llull
C/Quatre Camins, 2. 08022 Barcelona
{xavis,jclaudi,falias}@salle.url.edu

Resumen: La combinación de múltiples clasificadores no supervisados proporciona una vía hacia la construcción de sistemas robustos de clasificación de documentos. Este trabajo se centra en la combinación de clusterizadores difusos, proponiendo para ello dos funciones de consenso basadas en las estrategias de votación posicional de Borda y Condorcet. Los experimentos realizados sobre dos colecciones de documentos revelan que las funciones de consenso propuestas son capaces de obtener particiones por consenso de calidad comparable o superior a las obtenidas por técnicas pertenecientes al estado del arte, aunque su complejidad computacional es superior, lo que se debe al proceso de *ranking* implícito en las técnicas de votación posicional.
Palabras clave: Clustering de documentos, clustering difuso, clustering por consenso, votación de Borda, votación de Condorcet

Abstract: The combination of multiple clustering processes provides a means for building robust document clustering systems. This work focuses on the consolidation of fuzzy clusterings, proposing two consensus functions for soft cluster ensembles based on the Borda and Condorcet positional voting strategies. Experiments conducted on two document corpora reveal that the proposed soft consensus functions are capable of yielding consensus partitions of comparable or superior quality to those obtained by state-of-the-art clustering combiners, although their computational complexity is higher due to the ranking process inherent to positional voting techniques.
Keywords: Document clustering, fuzzy clustering, consensus clustering, Borda voting, Condorcet voting

1 Introduction

The development and expansion of the information and communication technologies, despite providing enormous quantities of information on a silver plate, pose a serious challenge to human analytic capabilities, not only by the large volumes of data available, but also by its growing complexity. Provided that a large proportion of such data comes in the form of written text, it seems logical to highlight the importance of automatic tools that allow knowledge extraction from large textual data repositories, regardless of their domain (Hearst, 2006).

When it comes to extracting knowledge from a given text collection, one of the primary tasks one thinks of is *organization*: clearly, arranging the contents of a document repository according to some meaningful structure (or *taxonomy*) –whose shape can vary widely, e.g. from parent-child hierarchical trees to network schemes or simple group structures– helps to gain some perspective on it. In fact, organizing information is one of the most innate activities involved in human learning (Anderberg, 1973). The increasingly growing volumes of digital textual data available call for the development of tools capable of organizing document collections in a fully automatic manner, so that no expert supervision nor domain knowledge is required.

Regardless of the taxonomy's layout, most text corpora unsupervised organization tech-

ISSN: 1135-5948

niques are typically based on classifying documents into groups (or *clusters*) according to the degree of similarity of their contents —i.e. their goal is to place documents dealing with similar topics in the same group, while placing dissimilar documents in separate clusters. This task, known as unsupervised classification or *clustering*, allows to represent the presumably high number of documents contained in a text corpus by means of a smaller number of clusters. The cluster-based data model obtained may be of help, for example, for simplifying browsing through large document collections (Cutting et al., 1992) or for refining ambiguous queries input to retrieval systems (Käki, 2005).

However, the unsupervised nature of the clustering problem poses a severe challenge at the time of configuring a document clustering system (Jain, Murty, and Flynn, 1999). Indeed, the effectiveness of the clustering task (i.e. the obtention of a *meaningful* partition of the text corpus subject to clustering) relies heavily on optimal (or quasi-optimal) decision-making with regard to the selection of *i)* the number of clusters documents are grouped into, *ii)* the way documents are represented, and *iii)* the clustering algorithm applied —see (Sevillano et al., 2007) for experimental evidences on the effect of the two last *clustering indeterminacies* on the quality of clustering processes results. Unfortunately, these important decisions are often made blindly unless deep domain knowledge is available, an assumption that hardly holds in practice[1] (Hearst, 2006). Therefore, entrusting the obtention of the partition of the document collection to a (possibly blindly configured) single clustering system seems to be a suboptimal way to proceed.

For this reason, our approach for obtaining clustering results which are *robust* to these indeterminacies follows a strategy that is rather the opposite: the document clustering practitioner is encouraged to use all the clustering systems at hand, compiling the resulting clusterings into a *cluster ensemble*, upon which a *consensus clustering* is derived by means of the application of a *consensus function*. The more similar the consensus clustering is to the highest quality cluster ensemble component, the greater robustness to clustering indeterminacies is achieved. The rationale of this approach is based on the fact that if the distinct clustering systems disagree, combining their outcomes may offer additional information and discriminatory power, thus obtaining a combined clustering that is closer to a hypothetical true classification (or *ground truth*) (Pinto et al., 2007). In other words, consensus clustering constitutes the unsupervised counterpart of supervised classifier committees, as it also aims to improve the quality of the component classifiers by combination (Dietterich, 2000). However, the inherent ambiguity of clusters identification (due to the unsupervised nature of the clustering problem) makes consensus clustering a more challenging task.

Most of the literature on consensus clustering is focused on consolidating the outcomes of multiple hard (or crisp) clustering systems, i.e. those that assign each document to a single cluster (Strehl and Ghosh, 2002; Fred and Jain, 2005). In contrast, fewer efforts have been conducted towards the combination of soft (or fuzzy) clustering processes, that is, those associating each document to each cluster to a certain degree (Dimitriadou, Weingessel, and Hornik, 2002; Punera and Ghosh, 2007). In our opinion, however, soft consensus clustering is an alternative worth considering, as crisp clustering is in fact a simplification of fuzzy clustering (i.e. a hard partition is always obtainable from a soft one by simply assigning each document to the cluster it is most strongly associated to) —a simplification that may give rise to the loss of valuable information, as a document may belong to more than one thematic cluster.

Allowing for this fact, this paper presents two consensus functions based on the Borda and the Condorcet positional voting strategies (Borda, 1781; Condorcet, 1785) for combining the outcomes of multiple fuzzy clustering systems. In this work, we describe our proposals and compare them to several state-

[1] Determining which is the 'correct' number of clusters in a data set is a tricky question, as equally satisfying (though substantially different) partitions of the same data can be obtained with different numbers of clusters, depending on the scale at which it is inspected (Chakaravathy and Ghosh, 1996). However, there exist several techniques for determining the most suitable number of groups a document collection should be clustered into, such as relative cluster validity indices (Halkidi, Batistakis, and Vazirgiannis, 2002) or optimization of criterion functions such as the Bayesian Inference Criterion (Schwarz, 1978). In this work, we make the simplifying –though not unusual– assumption that the desired number of clusters is known.

of-the-art soft consensus functions, both in terms of their computational cost and the quality of the consensus partitions they yield.

This paper is organized as follows: firstly, section 2 defines the concept of soft cluster ensembles, reviewing related work in the area of soft clustering systems combination. Next, section 3 presents a description of the two proposed consensus functions. In section 4, the performance of our positional voting based proposals is experimentally evaluated. Finally, section 5 presents the conclusions of this work.

2 Cluster ensembles and consensus functions

As mentioned earlier, the goal of consensus clustering is to merge the outcomes of l clustering systems into a single consensus partition. The two key elements of this process are the cluster ensemble $\mathbf{E}$ and the consensus function $\mathcal{F}$. The former is nothing but the compilation of the partitions yielded by the l clustering systems subject to combination, while the latter is the algorithm responsible for generating the consensus partition upon the cluster ensemble. Quite obviously, the features of both the cluster ensemble and the consensus function will vary depending on whether a crisp or a fuzzy consensus clustering scenario is considered. In the following paragraphs, we formally define soft cluster ensembles and enumerate some of the previous research efforts in the field of fuzzy clusterings combination.

In this work, we assume that the text collection under analysis contains n documents, and that the l clustering systems subject to combination partition the corpus into k clusters. Hence, a soft cluster ensemble $\mathbf{E}$ is formally defined as a $kl \times n$ real valued matrix resulting from the vertical concatenation of l matrices $\mathbf{\Lambda}_i$ of size $k \times n$. The ith of these matrices (which is referred to as the ith component of the cluster ensemble $\mathbf{E}$) is the outcome of the ith fuzzy clustering system subject to combination. Thus, the (a,b)th entry of matrix $\mathbf{\Lambda}_i$ represents the degree of association between the ath cluster and the bth document according to the ith clusterer. For convenience, let us define $\mathbf{\Lambda}_i$ in terms of its columns, represented by vectors $\boldsymbol{\lambda}_{ij}$ ($\forall i = 1, \ldots, l$ and $\forall j = 1, \ldots, n$) (see equation 1).

$$\mathbf{E} = \begin{pmatrix} \mathbf{\Lambda}_1 \\ \mathbf{\Lambda}_2 \\ \vdots \\ \mathbf{\Lambda}_l \end{pmatrix} \text{ where } \mathbf{\Lambda}_i = \begin{pmatrix} \boldsymbol{\lambda}_{i1} & \boldsymbol{\lambda}_{i2} & \ldots & \boldsymbol{\lambda}_{in} \end{pmatrix} \tag{1}$$

As far as the derivation of consensus functions for combining fuzzy clustering systems is concerned, the most relevant research efforts found in the literature are the voting-merging algorithm (Dimitriadou, Weingessel, and Hornik, 2002) or the consensus functions based on hypergraph partitioning and information-theoretic k-means (Punera and Ghosh, 2007). However, some consensus functions originally designed for combining crisp partitions, such as evidence accumulation (Fred and Jain, 2005), can be easily adapted for consolidating soft partitions (see (Sevillano, Alías, and Socoró, 2007)).

3 Soft consensus functions

In this section, two consensus functions for soft cluster ensembles based on positional voting methods are proposed. For starters, their rationale is described next.

Recall that each soft clustering system outputs the degree of association of each of the n documents to all the k clusters (in the shape of a $k \times n$ matrix $\mathbf{\Lambda}_i$, $\forall i \in [1, l]$). Therefore, the problem of combining the outcomes of multiple soft clustering processes by means of voting strategies requires interpreting the contents of $\mathbf{\Lambda}_i$ as the preference of the ith clustering system (which is regarded as a voter) for each cluster (or candidate). The voting procedure is responsible for consolidating the l voters' preferences, obtaining, as a result of an election, the preferences for each cluster regarding the classification of each document. For this reason, voting methods capable of dealing with voters' preferences constitute the basis of our consensus functions.

However, care must be taken at voting time, as the degree of association between documents and clusters may be directly or inversely proportional to the strength of this association (e.g. the elements of $\mathbf{\Lambda}_i$ may indistinctly be cluster membership probabilities or distances to cluster centroids, depending on the nature of each one of the l clustering systems subject to combination). For this reason, our consensus functions make use of

positional voting strategies, which are based on *ranking* the candidates (i.e. the clusters) according to the degree of confidence expressed by the voters (clustering systems) —thus, problems in scaling the voters confidence scores are avoided, although fine-grain information regarding preference differences between candidates is ignored (van Erp, Vuurpijl, and Schomaker, 2002).

Nevertheless, prior to the application of any voting strategy on soft cluster ensembles, there is a crucial problem to be solved. Due to the unsupervised nature of the clustering problem, clusters (i.e. candidates) are ambiguously identified. Therefore, it is necessary to perform a cluster alignment (or disambiguation) process between the l components of the cluster ensemble before voting (see section 3.1). Notice that this is not an issue of concern when applying voting strategies on supervised classifier ensembles, as categories are univocally defined in that case.

3.1 Cluster disambiguation

As just pointed out, the symbolic nature of cluster labels makes that a single clustering solution can be expressed by multiple equivalent representations. For instance, the two crisp clusterings $\boldsymbol{\lambda}_1 = [1\ 1\ 1\ 2\ 2\ 2\ 3\ 3\ 3]$ and $\boldsymbol{\lambda}_2 = [3\ 3\ 3\ 1\ 1\ 1\ 2\ 2\ 2]$ represent exactly the same partition of a toy corpus containing $n = 9$ documents into $k = 3$ clusters. This clusters ambiguity also affects soft clustering, as any permutation of the rows of a soft clustering matrix $\boldsymbol{\Lambda}$ gives rise to an equivalent fuzzy partition. Quite obviously, cluster identification ambiguity becomes a problem when voting is to be conducted among the l clustering solutions compiled in a cluster ensemble $\mathbf{E}$. This calls for the application of a technique capable of solving the cluster relabeling problem —an instance of the cluster disambiguation problem in which a one to one correspondence between clusters is considered (as in this work all the clusterings in the ensemble are assumed to have the same number of clusters, namely k).

To solve the cluster re-labeling problem we make use of the Hungarian method[2] (Kuhn, 1955), a technique that, given a pair of clustering solutions with k clusters each, is capable of finding, among the $k!$ possible cluster permutations, the one that maximizes the overlap between them in $O(k^3)$ time.

Given a cluster ensemble $\mathbf{E}$ containing l soft clusterings, the cluster disambiguation process consists in, taking one of them as a reference, apply the Hungarian method sequentially on the remaining $l-1$ components. As a result, a cluster aligned version of the cluster ensemble is obtained, and voting can be readily conducted on it.

3.2 Positional voting

In this work, we employ two positional voting strategies for deriving the consensus clustering solution. Both voting methods, called Borda and Condorcet voting –which date from the French revolution period– were devised for addressing the shortcomings of simple majority voting between more than two candidates (Borda, 1781; Condorcet, 1785), and they constitute the core of the two consensus functions described next.

3.2.1 BordaConsensus

The Borda voting method computes the mean rank of each candidate over all voters, re-ranking them according to their mean rank. This process results in a grading of all the k clusters with respect to each of the n documents, which is embodied in a $k \times n$ Borda voting matrix $\mathbf{B_E}$. Such grading process is conducted as follows: firstly, for each document (election), clusters (candidates) are ranked according to their degree of association with respect to it (from the most to the least strongly associated). Then, the top ranked candidate receives k points, the second ranked cluster receives $k-1$ points, and so on. After iterating this procedure across the l cluster ensemble components, the grading matrix $\mathbf{B_E}$ is obtained. The whole process is described in algorithm 1. Notice that the `Rank` procedure orders the clusters from the most to the least strongly associated to each document, yielding a ranking vector $\mathbf{r}$ which is a list of the k clusters ordered according to their degree of association with respect to the document under consideration (i.e. its first component, $\mathbf{r}(1)$, identifies the most strongly associated cluster, and so on). Thus, the `Rank` procedure must take into account whether the scalar values contained in $\boldsymbol{\lambda}_{ab}$ are directly or inversely proportional to the strength of association between documents and clusters.

Notice that the higher the value of the (i,j)th entry of $\mathbf{B_E}$, the more likely the jth

[2] In this work, we have employed the implementation of the Hungarian algorithm of (Buehren, 2008).

Input: Soft cluster ensemble $\mathbf{E}$
containing l fuzzy clusterings
$\mathbf{\Lambda}_i$ $(\forall i = 1 \dots l)$
Output: Borda voting matrix $\mathbf{B_E}$
Data: k clusters, n documents

```
Hungarian (E)
B_E = 0^{k×n}
for a = 1...l do
    for b = 1...n do
        r = Rank (λ_ab);
        for c = 1...k do
            B_E (r(c), b) =
            B_E (r(c), b) + (k − c + 1);
        end
    end
end
```

Algorithm 1: Symbolic description of the soft consensus function BordaConsensus. `Hungarian` and `Rank` are symbolic representations of the cluster disambiguation and cluster ordering procedures, respectively, while the vector $\boldsymbol{\lambda}_{ab}$ represents the bth column of the ath cluster ensemble component $\mathbf{\Lambda}_a$, $\mathbf{r}$ is a clusters ranking vector and $\mathbf{0}^{k\times n}$ represents a $k \times n$ zero matrix.

document belongs to the ith cluster. Thus, normalizing each column of matrix $\mathbf{B_E}$ (e.g. dividing each element by its column's L1-norm or applying a softmax normalization) transforms it into a cluster membership probability matrix. Moreover, assigning each document to the cluster it is most strongly associated to –breaking ties randomly– yields a crisp consensus clustering $\boldsymbol{\lambda}_c$.

3.2.2 CondorcetConsensus

Although often deemed as a multi-step unweighed voting algorithm, the Condorcet voting method can also be regarded as a positional voting strategy, as it employs the voters' preference choices between any given pair of candidates (van Erp, Vuurpijl, and Schomaker, 2002). In particular, this voting method performs an exhaustive pairwise candidate ranking comparison across voters, assigning one point to the winner of each one of these one-to-one confrontations. The result of this process is the Condorcet score matrix $\mathbf{C_E}$, the (i,j)th element of which indicates how many candidates does the ith candidate beat in one-to-one comparisons in the jth election (where candidates are clusters and an election corresponds to the clusterization

Input: Soft cluster ensemble $\mathbf{E}$
containing l fuzzy clusterings
$\mathbf{\Lambda}_i$ $(\forall i = 1 \dots l)$
Output: Condorcet voting matrix $\mathbf{C_E}$
Data: k clusters, n documents

```
Hungarian (E)
for b = 1...n do
    M = 0^{k×k}
    for a = 1...l do
        r = Rank (λ_ab);
        for c = 1...k do
            M (r(c), r(c+1,...,k)) =
            M (r(c), r(c+1,...,k)) + 1
        end
    end
    for c = 1...k do
        C_E (c, b) = Count
        (M(c, 1...k) ≥ l/2)
    end
end
```

Algorithm 2: Symbolic description of the soft consensus function CondorcetConsensus. `Hungarian` and `Rank` are symbolic representations of the cluster disambiguation and cluster ordering procedures, respectively, while the vector $\boldsymbol{\lambda}_{ab}$ represents the bth column of the ath cluster ensemble component $\mathbf{\Lambda}_a$, $\mathbf{r}$ is a clusters ranking vector and $\mathbf{0}^{k\times k}$ represents a $k \times k$ zero matrix.

of a document).

Algorithm 2 presents a description of the CondorcetConsensus consensus function. As in BordaConsensus, the `Rank` procedure must take into account whether the scalar values contained in $\boldsymbol{\lambda}_{ab}$ are directly or inversely proportional to the strength of association between documents and clusters. In each election, the (i,j)th entry of the square matrix $\mathbf{M}$ (usually referred to as the Condorcet sum matrix) counts the number of times the ith cluster is preferred over the jth one. The `Count` procedure is used for counting the number of elements of the cth row of matrix $\mathbf{M}$ that are greater or equal than $\frac{l}{2}$, which means that at least half of the voters preferred one candidate over another. Quite obviously, the Condorcet score matrix $\mathbf{C_E}$ can also be converted into a cluster membership matrix or transformed into a crisp consensus clustering $\boldsymbol{\lambda}_c$.

4 Experiments

4.1 Document collections

The experiments presented in this paper have been conducted on the miniNewsgroups (miniNG for short) and BBC text corpora.

The miniNG document collection[3] is a reduced version of the 20 Newsgroups text data set that contains $n = 2000$ articles posted in Usenet, each belonging to one of $k = 20$ predefined thematic classes (e.g. sci.electronics, rec.sport.baseball or talk.politics.mideast). The removal of stop words and of terms appearing in less than 5 documents gives rise to a bag-of-words representation of each article on a $d = 6679$ dimensional *tfidf*-weighted term space.

The BBC document corpus has been obtained from the online repository of the Machine Learning Group of the University College Dublin[4]. It consists of $n = 2225$ documents from the BBC news website corresponding to stories in $k = 5$ thematic areas (business, entertainment, politics, sport and tech). The original documents' representation used a 9636 dimensional term space which was reduced to $d = 6767$ real-valued attributes after removing those terms with a document frequency smaller than 5.

Additional document representations have been created by applying the following well-known feature extraction techniques: Principal Component Analysis, Independent Component Analysis, Non-negative Matrix Factorization and Random Projection. Besides providing diversity as far as data representation is concerned, these techniques are also employed with dimensionality reduction purposes. The reduced dimensionality of the resulting feature space is referred to as r, and it takes values in the interval $(3, d)$. Table 1 summarizes the main attributes of both text corpora.

4.2 Cluster ensembles generation

As regards the creation of the soft cluster ensemble components, we have employed the fuzzy c-means and the k-means clustering algorithms. Whereas the former is fuzzy by nature, the latter is not. However, some implementations of the k-means algorithm are capable of returning document to cluster centroid distances, which indeed constitute an indicator of the degree of association of documents to clusters. For the sake of greater algorithmic diversity, variants of k-means using the Euclidean, city block, cosine and correlation distances have been employed. As mentioned earlier, the desired number of clusters k to be found by these clustering algorithms has been set equal to the *real* number of categories in each collection.

	miniNG	**BBC**
Number of documents n	2000	2225
Number of classes k	20	5
Original term space dimensionality d	6679	6767
Reduced term space dimensionality r	[50:20:390]	[3:50:500]

Table 1: Description of the two document collections employed in this work. The notation $[a{:}b{:}c]$ denotes a sweep from a to c in steps of b

Applying these five clustering algorithms on distinct document representation has given rise to soft cluster ensembles of size $l = 365$ for the miniNG collection and $l = 285$ for the BBC corpus. Furthermore, in order to obtain a representative analysis of the consensus functions performance, besides using the cluster ensemble of size l, we have also generated cluster ensembles of sizes $\lfloor \frac{l}{20} \rfloor$, $\lfloor \frac{l}{10} \rfloor$, $\lfloor \frac{l}{5} \rfloor$ and $\lfloor \frac{l}{2} \rfloor$ (where $\lfloor x \rfloor$ denotes the application of the floor function on x), which are created by randomly picking a subset of the original cluster ensemble components. For each distinct cluster ensemble, ten independent runs of each consensus function are executed so as to obtain reliable results.

4.3 Compared consensus functions

In the upcoming experiments, BordaConsensus (BC) and CondorcetConsensus (CC) have been compared to several state-of-the-art consensus functions. This comparison is conducted against *i)* one of the pioneering soft consensus functions, called VMA (for Voting Merging Algorithm), which is based on solving the cluster correspondence problem on pairs of cluster ensemble components, and simultaneously applying a weighted version of the sum rule confidence voting method (Dimitriadou, Weingessel, and Hornik, 2002), and *ii)* the soft versions of four consen-

[3] Available at http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html

[4] Available at http://mlg.ucd.ie/content/view/21/

sus functions originally designed for combining crisp clusterings, namely the CSPA, HGPA and MCLA hypergraph-based consensus functions (Strehl and Ghosh, 2002), and the evidence accumulation (EAC) consensus function (Fred and Jain, 2005) —see (Sevillano, Alías, and Socoró, 2007; Punera and Ghosh, 2007) for details on their derivation.

4.4 Consensus functions evaluation

Both the computational cost of the compared consensus functions and the quality of the consensus clustering solutions they yield is evaluated. With regard to the former aspect, the CPU time required for their execution under Matlab 7.0.4 on a Dual Pentium 4/3GHz/1 GB RAM computer is measured.

As far as the evaluation of the quality of the consensus partitions is concerned, despite the proposed consensus functions output fuzzy consensus clusterings, we have compared their *hardened* version $\boldsymbol{\lambda}_c$ with respect to the ground truth $\boldsymbol{\gamma}$ of each document collection in terms of normalized mutual information $\phi^{(\mathrm{NMI})}$ (see equation 2).

$$\phi^{(\mathrm{NMI})}(\boldsymbol{\gamma}, \boldsymbol{\lambda}_c) = \frac{2}{n} \sum_{l=1}^{k} \sum_{h=1}^{k} n_{h,l} \log_{k^2} \left(\frac{n_{h,l}\, n}{n_h^{(\boldsymbol{\gamma})} n_l^{(\boldsymbol{\lambda}_c)}} \right) \quad (2)$$

where $n_h^{(\boldsymbol{\gamma})}$ is the number of documents in cluster h according to $\boldsymbol{\gamma}$, $n_l^{(\boldsymbol{\lambda}_c)}$ is the number of documents in cluster l according to $\boldsymbol{\lambda}_c$, $n_{h,l}$ denotes the number of documents in cluster h according to $\boldsymbol{\gamma}$ as well as in group l according to $\boldsymbol{\lambda}_c$, n is the number of documents in the corpus, and k is the number of clusters into which documents are clustered according to $\boldsymbol{\lambda}_c$ and $\boldsymbol{\gamma}$ (Strehl and Ghosh, 2002).

The reason for using this evaluation scheme is threefold: firstly, a fuzzy ground truth is not available for these data sets, so fuzzy consensus clusterings cannot be directly evaluated. Secondly, provided that the CSPA, HGPA, MCLA and EAC consensus functions output hard consensus clusterings, fair consensus function comparison requires converting the soft consensus clusterings output by VMA, BC and CC to crisp consensus labelings $\boldsymbol{\lambda}_c$ —which simply boils down to assigning each document to the cluster it is more strongly associated to. And thirdly, $\phi^{(\mathrm{NMI})}$ is theoretically well-founded, unbiased, symmetric with respect to $\boldsymbol{\lambda}_c$ and $\boldsymbol{\gamma}$, plus being normalized in the $[0, 1]$ interval —the higher the value of $\phi^{(\mathrm{NMI})}$, the more similar $\boldsymbol{\lambda}_c$ and $\boldsymbol{\gamma}$ are (Strehl and Ghosh, 2002).

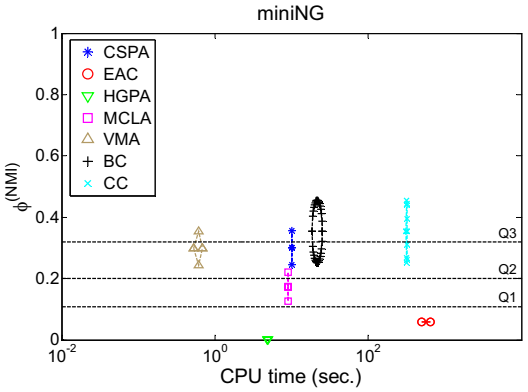

Figure 1: $\phi^{(\mathrm{NMI})}$ vs. CPU time mean ± 2-standard deviation regions of the soft consensus functions on the miniNG corpus

4.5 Experimental results

Figure 1 shows a quality ($\phi^{(\mathrm{NMI})}$ with respect to the ground truth) vs. time complexity (CPU running time measured in seconds) diagram that describes, in a summarized manner, the performance of the seven soft consensus functions compared on the miniNG text collection. For each consensus function, the depicted scatterplot corresponds to the region limited by the mean ± 2-standard deviation curves corresponding to the magnitude associated to each axis. Moreover, so as to compare the quality of the consensus clusterings to that of the soft cluster ensemble components, the dashed horizontal lines appearing in figure 1 correspond to the lower quartile (Q1), median (Q2), and upper quartile (Q3) of the $\phi^{(\mathrm{NMI})}$ values of the latter.

The analysis of figure 1 reveals that the BC and CC consensus functions yield consensus clusterings of higher quality than most of the soft cluster ensemble components, besides reaching higher $\phi^{(\mathrm{NMI})}$ scores than all the state-of-the-art consensus functions employed —in fact, crossed pairwise t-tests indicate that the observed inter-consensus functions $\phi^{(\mathrm{NMI})}$ differences are statistically significant at a >99% confidence level.

From a computational complexity perspective, the proposed consensus functions are only faster than EAC. Notice that VMA is clearly the fastest alternative, due to the simultaneity of the cluster disambiguation and

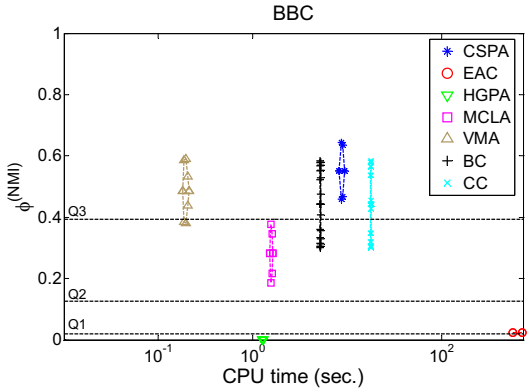

Figure 2: $\phi^{(\mathrm{NMI})}$ vs. CPU time mean ± 2-standard deviation regions of the soft consensus functions on the BBC corpus

voting processes in this consensus function.

The results of the soft consensus clustering experiments conducted on the BBC corpus are depicted in the $\phi^{(\mathrm{NMI})}$ vs. CPU time diagram of figure 2. Notice that VMA is again the fastest consensus function. The proposed consensus functions (BC and CC) are slower than MCLA and HGPA, the latter being also slower than CSPA, while the former is faster. All the pairwise running time differences measured are statistically significant at a 95% level.

As regards the quality of the consensus clustering solutions obtained, the BordaConsensus and CondorcetConsensus clustering combiners deliver pretty good performances, being *i)* far better than EAC and HGPA, *ii)* notably better than MCLA, *iii)* statistically equivalent to VMA, and *iv)* only inferior to CSPA.

5 Conclusions

The BordaConsensus and CondorcetConsensus consensus functions constitute pioneering efforts as far as the use of positional voting methods in fuzzy consensus clustering is concerned. As mentioned earlier, this type of voting naturally lends itself to the combination of soft clusterings of different natures, as it avoids scaling voters' confidence scores. However, the application of voting methods for combining clustering systems is not new: for instance, unweighed voting strategies (van Erp, Vuurpijl, and Schomaker, 2002) such as plurality and majority voting have been applied for deriving consensus clustering solutions on hard cluster ensembles (Dudoit and Fridlyand, 2003; Fischer and Buhmann, 2003). To our knowledge, the only voting-based consensus function for soft cluster ensembles is the Voting-Merging Algorithm (VMA) of (Dimitriadou, Weingessel, and Hornik, 2002), which employs a weighted version of the sum rule for confidence voting. All these algorithms use the Hungarian method for solving the cluster correspondence problem, as our proposals do.

The comparative performance analysis of the two proposed consensus functions has revealed that they constitute a feasible alternative for conducting consensus clustering on soft cluster ensembles, as they are capable of yielding consensus partitions of comparable or superior quality to those obtained by state-of-the-art clustering combiners. An additional appealing feature of our proposals is that they naturally deliver fuzzy consensus clustering solutions, which makes all sense in a soft clustering scenario —a fact other recent consensus functions for soft cluster ensembles do not consider (Punera and Ghosh, 2007).

As regards the computational complexity comparison, the execution of BC and CC –specially the latter– is more costly than most of the implemented state-of-the-art consensus functions. This is due to the sequential application of the cluster disambiguation and the voting processes that, added to the transformation of document to cluster associations into rankings, increases the execution time of our proposals. For this reason, we plan to apply the simultaneous cluster alignment plus voting strategy employed by the VMA consensus function in BordaConsensus and CondorcetConsensus, as we consider that will surely reduce their execution time without significantly reducing the quality of the consensus clusterings obtained. Furthermore, we also intend to *i)* evaluate alternatives to the Hungarian cluster disambiguation method, as far as their impact on the quality of the consensus clusterings and on the computational complexity is concerned, and *ii)* develop versions of the BC and CC consensus functions capable of consolidating fuzzy partitions with distinct number of clusters.

References

Anderberg, M.R. 1973. *Cluster Analysis for Applications*. Monographs and Textbooks on Probability and Mathematical Statistics. Academic Press, Inc., New York.

Borda, J.C. de. 1781. *Memoire sur les Elections au Scrutin*. Histoire de l'Academie Royale des Sciences, Paris.

Buehren, M. 2008. Functions for the rectangular assignment problem. http://www.mathworks.com/matlabcentral/fileexchange/6543.

Chakaravathy, S.V. and J. Ghosh. 1996. Scale based clustering using a radial basis function network. *IEEE Transactions on Neural Networks*, 2(5):1250–1261.

Condorcet, M. de. 1785. *Essai sur l'application de l'analyse la probabilit des decisions rendues la pluralit des voix*.

Cutting, D.R., D.R. Karger, J.O. Pedersen, and J.W. Tukey. 1992. Scatter/Gather: a cluster-based approach to browsing large document collections. In *Proc. 15th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318–329.

Dietterich, T.G. 2000. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems, LNCS vol. 1857*, pages 1–15. Springer.

Dimitriadou, E., A. Weingessel, and K. Hornik. 2002. A combination scheme for fuzzy clustering. *Int'l Journal of Pattern Recognition and Artificial Intelligence*, 16(7):901–912.

Dudoit, S. and J. Fridlyand. 2003. Bagging to Improve the Accuracy of a Clustering Procedure. *Bioinformatics*, 19(9):1090–1099.

Fischer, B. and J.M. Buhmann. 2003. Bagging for path-based clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1411–1415.

Fred, A. and A.K. Jain. 2005. Combining Multiple Clusterings Using Evidence Accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):835–850.

Halkidi, M., Y. Batistakis, and M. Vazirgiannis. 2002. Cluster Validity Methods: Part II. *ACM SIGMOD Record*, 31(3):19–27.

Hearst, M.A. 2006. Clustering versus faceted categories for information exploration. *Communications of the ACM*, 49(4):59–61.

Jain, A.K., M.N. Murty, and P.J. Flynn. 1999. Data Clustering: a Survey. *ACM Computing Surveys*, 31(3):264–323.

Käki, M. 2005. Findex: search result categories help users when document ranking fails. In *Proc. ACM SIGCHI Int'l Conference on Human Factors in Computing Systems*, pages 131–140. ACM Press.

Kuhn, H. 1955. The Hungarian Method for the Assignment Problem. *Naval Research Logistic Quarterly*, 2:83–97.

Pinto, F.R., J.A. Carrio, M. Ramirez, and J.S. Almeida. 2007. Ranked Adjusted Rand: integrating distance and partition information in a measure of clustering agreement. *BMC Bioinformatics*, 8(44):1–13.

Punera, K. and J. Ghosh. 2007. Soft Consensus Clustering. In J. Oliveira and W. Pedrycz, editors, *Advances in Fuzzy Clustering and its Applications*, pages 69–92. Wiley.

Schwarz, G. 1978. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464.

Sevillano, X., F. Alías, and J.C. Socoró. 2007. BordaConsensus: a New Consensus Function for Soft Cluster Ensembles. In *Proc. 30th ACM SIGIR Conference*, pages 743–744.

Sevillano, X., G. Cobo, F. Alías, and J.C. Socoró. 2007. Text clustering on latent thematic spaces: Variants, strenghts and weaknesses. In M.E. Davies, C.C. James, S.A. Abdallah, and M.D. Plumbley, editors, *Independent Component Analysis and Signal Separation, LNCS vol. 4666*, pages 794–801. Springer.

Strehl, A. and J. Ghosh. 2002. Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal on Machine Learning Research*, 3:583–617.

van Erp, M., L. Vuurpijl, and L. Schomaker. 2002. An overview and comparison of voting methods for pattern recognition. In *Proc. 8th Int'l Workshop on Frontiers in Handwriting Recognition*, pages 195–200.