Document Clustering Games in Static and Dynamic Scenarios

Rocco Tripodi[1] and Marcello Pelillo[1,2]

[1] ECLT, Ca' Foscari University, Ca' Minich, Venice, Italy,
[2] DAIS, Ca' Foscari University, Via Torino, Venice, Italy.
{rocco.tripodi,pelillo}@unive.it

Abstract. In this work we propose a game theoretic model for document clustering. Each document to be clustered is represented as a player and each cluster as a strategy. The players receive a reward interacting with other players that they try to maximize choosing their best strategies. The geometry of the data is modeled with a weighted graph that encodes the pairwise similarity among documents, so that similar players are constrained to choose similar strategies, updating their strategy preferences at each iteration of the games. We used different approaches to find the prototypical elements of the clusters and with this information we divided the players into two disjoint sets, one collecting players with a definite strategy and the other one collecting players that try to learn from others the correct strategy to play. The latter set of players can be considered as new data points that have to be clustered according to previous information. This representation is useful in scenarios in which the data are streamed continuously. The evaluation of the system was conducted on 13 document datasets using different settings. It shows that the proposed method performs well compared to different document clustering algorithms[3].

1 Introduction

Document clustering is a particular kind of clustering that involves textual data. It can be employed to organize tweets [24], news [4], novels [3] and medical documents [6]. It is a fundamental task in text mining and have different applications in document organization and language modeling [15].

State-of-the-art algorithms designed for this task are based on generative models [38], graph models [37,29] and matrix factorization techniques [35,20]. Generative models and topic models [5] aim at finding the underlying distribution that created the set of data objects, observing the sequences of objects and features. One problem with these approaches is the conditional-independence assumption that does not hold for textual data and in particular for streaming documents. In fact, streamed documents such as mails, tweets or news can be

[3] This paper will be published in the series Lecture Notes in Computer Science (LNCS) published by Springer, containing the ICPRAM 2016 best papers.

generated in response to past events, creating topics and stories that evolve over time.

CLUTO is a popular graph-based algorithm for document clustering [36]. It employs a graph to organize the documents and different criterion functions to partition this graph into a predefined number of clusters. The problem with partitional approaches is that these approaches require to know in advance the number of clusters into which the data points have to be divided. A problem that can be restrictive in real applications and in particular on streaming data.

Matrix factorization algorithms, such as Non-negative Matrix Factorization (NMF) [12,7], assume that words that occur together can represent the features that characterize a clusters. Ding et al. [7] demonstrated the equivalence between NMF and Probabilistic Latent Semantic Indexing, a popular technique for document clustering. Also with these approaches it is required to know in advance the number of clusters into which the data have to be organized.

A general problem, common to all these approaches, concerns the temporal dimension. In fact, for these approaches it is difficult to deal with streaming datasets. A non trivial problem, since in many real world applications documents are streamed continuously. This problem is due to the fact that these approaches operate on a dataset as a whole and need to be recomputed if the dataset changes. It can be relevant also in case of huge static datasets, because of scalability issues [1]. In these contexts an incremental algorithm would be preferable, since with this approach it is possible to cluster the data sequencially.

With our approach we try to overcome this problem. We cluster part of the data producing small clusters that at the beginning of the process can be considered as cluster representative. Then we cluster new instances according to this information. With our approach is also possible deal with situations in which the number of clusters is unknown, a common situation in real world applications. The clustering of new instances is defined as a game, in which there are labeled players (from an initial clustering), which always play the strategy associated to their cluster and unlabeled players that learn their strategy playing the games iteratively and obtaining a feedback from the strategy that their co-players are adopting.

In contrast to other stream clustering algorithm our approach is not based only on proximity relations, such as in methods based on partitioning representatives [2]. With these approaches the cluster membership of new data points is defined selecting the cluster of their closest representative. With our approach the cluster membership emerges dynamically from the interactions of the players and all the neighbors of a new data point contribute in different proportion to the final cluster assignment. It does not consider only local information to cluster new data points but find solutions that are globally consistent. In fact, if we consider only local information the cluster membership of a point in between two or more clusters could be arbitrary.

The rest of this contribution is organized as follows. In the next Section, we briefly introduce the basic concepts of classical game theory and evolutionary game theory that we used in our framework; for a more detailed analysis of these topics the reader is referred to [34,13,23]. Then we introduce the *dominant set* clustering algorithm [18,21] that we used in part of our experiments to find the initial clustering of the data. In Section 4 we describe our model and in the last section we present the evaluation of our approach in different scenarios. First we use it to cluster static datasets and then, in Section 5.6, we present the evaluation of our method on streaming data. This part extends our previous work [32] and demonstrates that the proposed framework can be used in different scenarios with good performances.

2 Game Theory

Game theory was introduced by Von Neumann and Morgenstern [33]. Their idea was to develop a mathematical framework able to model the essentials of decision making in interactive situations. In its *normal-form* representation, which is the one we use in this work, it consists of a finite set of players $I = \{1, .., n\}$, a set of pure strategies, $S_i = \{s_1, ..., s_m\}$, and a utility function $u_i : S_1 \times ... \times S_n \rightarrow \mathbb{R}$ that associates strategies to payoffs; n is the number of players and m the number of pure strategies. The games are played among two different players and each of them have to select a strategy. The outcome of a game depends on the combination of strategies (strategy profile) played at the same time by the players involved in it, not just on the single strategy chosen by a player. For example we can consider the following payoff matrix,

$P_1 \backslash P_2$	strategy 1	strategy 2
strategy 1	-5,-5	0,-6
strategy 2	-6,0	-1,-1

Table 1. The payoff matrix of the prisoner's dilemma game.

where, for example, player 1 get -5 when he chooses strategy 1 and player 2 chooses strategy 1. Furthermore, in *non-cooperative games* the players choose their strategies independently, considering what the other players can play and try to find the best strategy profile to employ in a game.

An important assumption in game theory is that the players try to maximize their utility in the games (u_i), selecting the strategies that can give the highest payoff, considering what strategies the other player can employ. The players try to find the strategies that are better than others regardless what the other player does. These strategies are called *strictly dominant* and can occur if and only if:

$$u(s_i^*, s_{-i}) > u_i(s_i, s_{-i}), \forall s_{-i} \in S_{-i} \tag{1}$$

where s_{-i} denotes the strategy chosen by the other player(s).

The key concept of game theory is the Nash equilibrium that is used to predict the outcome of a strategic interaction. It can be defined as those strategy profiles in which no player has the incentive to unilaterally deviate from it, because there is no way to do increment the payoff. The strategies in a Nash equilibrium are best responses to all other strategies in the game, which means that they give the most favorable outcome for a player, given other players' strategies.

The players can play *mixed strategies*, which are probability distributions over pure strategies. In this context, the players select a strategy with a certain probability. A mixed strategy set can be defined as a vector $x = (x^1, \dots, x^m)$, where m is the number of pure strategies and each component x^h denotes the probability that a particular player select its hth pure strategy. Each player has a strategy set that is defined as a standard simplex:

$$\varDelta = \Big\{ x \in \mathbb{R} : \sum_{h=1}^{m} x^h = 1, \text{ and } x^h \geq 0 \textit{ for all } h \Big\}. \tag{2}$$

A mixed strategy set corresponds to a point on the simplex δ, whose corners represent pure strategies.

A strategy profile can be defined as a pair (p, q) where $p \in \varDelta_i$ and $q \in \varDelta_j$. The payoff of this strategy profile is computed as:

$$u_i(p, q) = p \cdot A_i q \ , \ u_j(p, q) = q \cdot A_j p, \tag{3}$$

where A_i and A_j are the payoff matrices of player i and j respectively. The Nash equilibrium within this setting can be computed in the same way it is computed in pure strategies. In this case, it consists in a pair of mixed strategies such that each one is a best response to the other.

To overcome some limitations of traditional game theory, such as the hyper-rationality imposed on the players, a dynamic version of game theory was introduced. It was proposed by John Maynard Smith and George Price [26], as evolutionary game theory. Within this framework the games are not static and are played repeatedly. This reflect real life situations, in which the choices change according to past experience. Furthermore, players can change a behavior according to heuristics or social norms [28]. In this context, players make a choice that maximizes their payoffs, balancing cost against benefits [17].

From a machine learning perspective this process can be seen as an *inductive learning* process, in which agents play the games repeatedly and at each iteration of the system they update their beliefs on the strategy to take. The update is done considering what strategy has been effective and what has not in previous games. With this informatioin, derived from the observation of the payoffs obtained by each strategy, the players can select the strategy with higher payoff.

The strategy space of each players is defined as a mixed strategy profile x_i and the mixed strategy space of the game is given by the Cartesian product of all the players' strategy space:

$$\varTheta = \times_{i \in I} \varDelta_i. \tag{4}$$

The expected payoff of a strategy e^h in a single game is calculated as in mixed strategies (see Equation 3) but, in evolutionary game theory, the final payoff of each player is the sum of all the partial payoffs obtained during an iteration. The payoff corresponding to a single strategy is computed as:

$$u_i(e_i^h) = \sum_{j=1}^{n} (A_{ij} x_j)^h \tag{5}$$

and the average payoff is:

$$u_i(x) = \sum_{j=1}^{n} x_i^T A_{ij} x_j, \tag{6}$$

where n is the number of players with whom player i play the games and A_{ij} is the payoff matrix among player i and j. At each iteration a player can update his strategy space according to the payoffs gained during the games, it allocates more probability on the strategies with high payoff, until an equilibrium is reached, a situation in which it is not possible to obtain higher payoffs.

To find the Nash equilibrium of the system it is common to use the replicator dynamic equation [30],

$$\dot{x} = [u(e^h) - u(x)] \cdot x^h . \forall h \in x. \tag{7}$$

This equation allows better than average strategies to increase at each iteration. It can be used to analyze frequency-dependent selection processes [16], furthermore, the fixed points of equation 7 correspond to Nash equilibria [34]. We used the discrete time version of the replicator dynamic equation for the experiments of this work.

$$x^h(t+1) = x^h(t) \frac{u(e^h)}{u(x)} \;\; \forall h \in x(t). \tag{8}$$

The players update their strategies at each time step t considering the strategic environment in which they are playing.

The complexity of each step of the replicator dynamics is quadratic but there are more efficient dynamics that can be used, such as the *infection and immunization* dynamics that has a linear-time/space complexity per step and it is known to be as accurate as the replicator dynamics [22].

3 Dominant Set Clustering

Dominant set is a graph based clustering algorithm that generalizes the notion of maximal clique from unweighted undirected graphs to edge-weighted graphs [18,21]. With this algorithm it is possible to extract compact structures from a graph in an efficient way. Furthermore, it can be used on symmetric and asymmetric similarity graphs and does not require any parameter. With this framework it is possible to obtain measures of clusters cohesiveness and to evaluate the strength of participation of a vertex to a cluster. It models the well-accepted definition of a cluster, which states that a cluster should have high internal homogeneity and that there should be high inhomogeneity between the objects in the cluster and those outside [10].

The extraction of compact structures from graphs that reflect these two conditions, is given by the following quadratic form:

$$f(x) = x^T A x. \tag{9}$$

Where A is a similarity graph and x is a probability vector, whose components indicate the participation of each node of the graph to a cluster. In this context, the clustering task corresponds to the task of finding a vector x that maximizes f and this can be done with the following program:

$$\begin{aligned} \text{maximize } & f(x) \\ \text{subject to } & x \quad \in \quad \varDelta. \end{aligned} \tag{10}$$

Where $\varDelta$ represents the standard simplex. A (local) solution of program (10) corresponds to a maximally cohesive structure in the graph [10].

The solution of program (10) can be found using the discrete time version of the replicator dynamic equation, computed as follows,

$$x(t+1) = x\frac{Ax}{x^T A x}, \tag{11}$$

where x represent the strategy space at time t.

The clusters are extracted sequentially from the graph using a peel-off strategy to remove the objects belonging to the extracted cluster, until there are no more objects to cluster or some predefined criteria are satisfied.

4 Document Clustering Games

In this section we present step by step our approach to document clustering. First we describe how the documents are represented and how we prepare the data and structure them using a weighted graph. Then we pass to the preliminary clustering in order to divide the data points in two disjoint sets of labeled and unlabeled players. With this information we can initialize the strategy space of the players and run the dynamics of the system that lead to the final clustering of the data.

4.1 Document representation

The documents of a datasets are processed with a *bag-of-words* (BoW) model. With this method each document is represented as a vector indexed according to the frequency of the words in it. To do this it is necessary to construct the vocabulary of the text collection that is composed by the set of unique words in the corpus. BoW represents a corpus with a *document-term matrix*. It consists in a $N \times T$ matrix M, where N is the number of documents in the corpus and T

the number of words in the vocabulary. The words are considered as the features of the documents. Each element of the matrix M indicates the frequency of a word in a document.

The BoW representation can lead to a high dimensional space, since the vocabulary size increases as sample size increases. Furthermore, it does not incorporate semantic information treating homonyms as the same feature. These problems can result in bad representations of the data and for this reason there where introduced different approaches to balance the importance of the features and also to reduce their number, focusing only on the most relevant.

An approach to weigh the importance of a feature is the *term frequency - inverse document frequency* (tf-idf) method [15]. This technique takes as input a document-term matrix M and update it with the following equation,

$$tf\text{-}idf(d,t) = tf(d,t) \cdot log\frac{D}{df(d,t)}, \tag{12}$$

where $df(d,t)$ is the number of documents that contain word t. Then the vectors are normalized to balance the importance of each feature.

Latent Semantic Analysis (LSA) is a technique used to infer semantic information [11] from a *document-term matrix*, reducing the number of features. Semantic information is obtained projecting the documents into a *semantic space*, where the relatedness of two terms is computed considering the number of times they appear in a similar context. Single Value Decomposition (SVD) is used to create an approximation of the *document-term matrix* or *tf-idf matrix.* It decomposes a matrix M in:

$$M = U\Sigma V^T, \tag{13}$$

where Σ is a diagonal matrix with the same dimensions of M and U and V are two orthogonal matrices. The dimensions of the feature space is reduced to k, taking into account only the first k dimensions of the matrices in Equation (13).

4.2 Data preparation

This new representation of the data is used to compute the pairwise similarity among documents and to construct the proximity matrix W, using the cosine distance as metric,

$$\cos\theta \frac{v_i \cdot v_j}{||v_i||||v_j||} \tag{14}$$

where the nominator is the intersection of the words in the vectors that represent two documents and $||v||$ is the norm of the vectors, which is calculated as: $\sqrt{\sum_{i=1}^{n} w_i^2}$.

4.3 Graph construction

W can be used to represent a text collection as a graph G, whose nodes represent documents and edges are weighted according to the information stored in W.

Since, the cosine distance acts as a linear kernel, considering only information between vectors under the same dimension, it is common to smooth the data using a kernel function and transforming the proximity matrix W into a similarity matrix S [25]. It can also transform a set of complex and nonlinearly separable patterns into linearly separable patterns [9]. For this task we used the Gaussian kernel,

$$s_{ij} = exp\left\{-\frac{w_{ij}^2}{\sigma^2}\right\} \tag{15}$$

where w_{ij} is the dissimilarity among pattern i and j and σ is a positive real number that determines the kernel width. This parameter is calculated experimentally, since it is not possible to know in advance the nature of the data and the clustering separability indices [19]. The data representation on the graph can be improved using graph Laplacian techniques. These techniques are able to decrease the weights of the edges between different clusters making them more distant. The normalized graph Laplacian is computed as $L = D^{-1/2}SD^{-1/2}$, where D is the degree matrix of S.

Another technique that can be used to better represent the data is sparsification, that consists in reducing the number of nodes in the graph, focusing only on the most important. This refinement is aimed at modeling the local neighborhood relationships among nodes and can be done with two different methods, the ϵ-neighborhood technique, which maintains only the edges which have a value higher than a predetermined threshold, ϵ; and the k-nearest neighbor technique, which maintains only the highest k values. It results in a similarity matrix that can be used as the adjacency matrix of a weighted graph G.

The effect of the processes described above is presented in Fig. 1. Near the main diagonal of the matrices it is possible to recognize some blocks which represent clusters. The values of those points are low in the cosine matrix, since it encodes the proximity of the points. Then the matrix is transformed into a similarity matrix by the Gaussian kernel, in fact, the values of the points near the main diagonal in this representation are high. It is possible to note that some noise was removed with the Laplacian matrix. The points far from the diagonal appear now clearer and the blocks are more compact. Finally the k-nn matrix remove many nodes from the representation, giving a clear picture of the clusters.

We used the Laplacian matrix L for the experiments with the *dominant set*, since it requires that the similarity values among the elements of a cluster are very close to each other. Graph G is used to run the clustering games, since this framework does not need a dense graph to cluster the data points.

4.4 Clustering

We use the *dominant set* algorithm to extract the prototypical elements of each cluster with two different settings, one in which we give as input the number of clusters to extract and the other without this information. In the fist case we

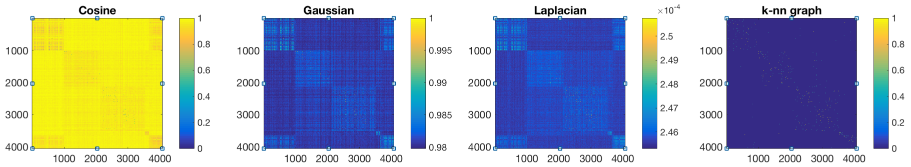

Fig. 1. Different data representations for a dataset with 5 classes of different sizes.

extract the first K clusters from a dataset and then run the document clustering games to cluster the remaining clusters. This situation can be interpreted as the case in which there are some labeled points in the data and new points have to be clustered according to this evidence. In the second case we run *dominant set* recursively to extract small clusters and then use the document clustering games to cluster the clusters, merging them according to their similarity. The similarity among two clusters C_i and C_j is computed as:

$$sim(C_i, C_j) = \frac{\sum_{r \in C_i} \sum_{t \in C_j} s_{rt}}{|C_i| + |C_j|} \tag{16}$$

We conducted also experiments in which we simulated the streaming data process. This is done dividing a dataset in random folds and clustering the dataset iteratively adding a fold at time to measure if the performances of the system are constant. In this case we used a fold (8% of the data) as initial clustering.

4.5 Strategy space implementation

The clustering phase serves as preliminary phase to partition the data into two disjoint sets, one containing clustered objects and the other unclustered. Clustered objects supply information to unclustered nodes in the graph. We initialized the strategy space of the player in these two sets as follows,

$$x_i^h = \begin{cases} K^{-1}, & \text{if node } i \text{ is unclustred.} \\ 1, & \text{if node } i \text{ is in cluster } h, \end{cases} \tag{17}$$

where K is the number of clusters to extract and K^{-1} ensures that the constraints required by a game theoretic framework are met (see equation (2)).

4.6 Clustering games

We assume that each player $i \in I$ that participates in the games is a document and that each strategy $s \in S_i$ is a particular cluster. The players can choose a determined strategy from their strategy space that is initialized as described in previous section and can be considered as a mixed strategy space (see Section

2). The games are played among two similar players, i and j. The payoff matrix among two players i and j is defined as an identity matrix of rank K, A_{ij}.

This choice is motivated by the fact that in this context all the players have the same number of strategies and in the studied contexts the number of clusters of each dataset is low. In works in which there are many interacting classes it is possible to use a similarity function to construct the payoff matrix, as described in [31].

The best choice for two similar players is to be clustered in the same cluster, this is imposed with the entry $A_{ij} = 1, i = j$. This kind of game is called *imitation game* because the players try to learn their strategy observing the choices of their co-players. For this reason the payoff function of each player is additively separable and is computed as described in Section 2. Specifically, in the case of clustering games there are labeled and unlabeled players that, as proposed in [8], can be divided in two disjoint sets, I_l and I_u. We have K disjoint subsets, $I_l = \{I_{l|1}, ..., I_{l|K}\}$, where each subset denotes the players that always play their hth pure strategy.

Only unlabeled players play the games, because they have to decide their best strategy (cluster membership). This strategy is selected taking into account the similarity that a player share with other players and the choices of these players. Labeled players act as bias over the choices of unlabeled players because they always play a defined strategy and unlabeled players influence each other. The players adapt to the strategic environment, gradually adjusting their preferences over strategies [23]. Once equilibrium is reached, the cluster of each player i, corresponds to the strategy, with the highest value.

The payoffs of the games are calculated with equations 5 and 6, which in this case, with labeled and unlabeled players, can be defined as,

$$u_i(e_i^h) = \sum_{j \in I_u} (g_{ij} A_{ij} x_j)^h + \sum_{h=1}^{K} \sum_{j \in I_{l|h}} (g_{ij} A_{ij})^h \tag{18}$$

and,

$$u_i(x) = \sum_{j \in I_u} x_i^T g_{ij} A_{ij} x_j + \sum_{k=1}^{K} \sum_{j \in I_{l|h}} x_i^T (g_{ij} A_{ij})^h. \tag{19}$$

where the first part of the equations calculates the payoffs that each player obtains from unclustered players and the second part computes the payoffs obtained from labeled players. The Nash equilibria of the system are calculated the replicator dynamics equation 8.

5 Experimental Setup

The performances of the systems are measured using the accuracy (AC) and the normalized mutual information (NMI). AC is calculated with the following equation,

$$AC = \frac{\sum_{i=1}^{n} \delta(\alpha_i, map(l_i))}{n}, \tag{20}$$

where n denotes the total number of documents in the dataset and $\delta(x, y)$ is equal to 1 if x and y are clustered in the same cluster. the function $map(L_i)$ maps each cluster label l_i to the equivalent label in the benchmark, aligning the labeling provided by the benchmark and those obtained with our clustering algorithm. It is done using the Kuhn-Munkres algorithm [14]. The NMI was introduced by Strehl and Ghosh [27] and indicates the level of agreement between the clustering C provided by the ground truth and the clustering C' produced by a clustering algorithm. This measure takes into account also the partitioning similarities of the two clustering and not just the number of correctly clustered objects. The mutual information (MI) between the two clusterings is computed with the following equation,

$$MI(C, C') = \sum_{c_i \in C, c'_j \in C'} p(c_i, c'_j) \cdot log_2 \frac{p(c_i, c'_j)}{p(c_i) \cdot p(c'_j)}, \tag{21}$$

where $p(c_i)$ and $p(c'_i)$ are the probabilities that a document belongs to cluster c_i and c'_i, respectively; $p(c_i, c'_i)$ is the probability that the selected document belongs to c_i as well as c'_i at the same time. The MI information is then normalized with the following equation,

$$NMI(C, C') = \frac{MI(C, C')}{max(H(C), H(C'))} \tag{22}$$

where $H(C)$ and $H(C')$ are the entropies of C and C', respectively, This measure ranges from 0 to 1. It is equal to 1 when the two clustering are identical and it is equal to 0 if the two sets are independent. We run each experiment 50 times and present the mean results with standard deviation (±).

We evaluated our model on the same datasets[4] used in [38]. In that work it has been conducted an extensive comparison of different document clustering algorithms. The description of these datasets is shown in Table 2. The authors used 13 datasets (described in Table 2). The datasets have different sizes (n_d), from 204 documents (tr23) to 8580 (*sports*). The number of classes (K) is also different and ranges from 3 to 10. Another important feature of the datasets is the size of the vocabulary (n_w) of each dataset that ranges from 5832 (*tr23*) to 41681 (*classic*) and is function of the number of documents in the dataset, their size and the number of different topics in it, that can be considered as clusters. The datasets are also described with n_c and *Balance*. n_c indicates the average number of documents per cluster and *Balance* is the ratio among the size of the smallest cluster and that of the largest.

[4] `http://www.shi-zhong.com/software/docdata.zip` .

Data	n_d	n_v	K	n_c	Balance
NG17-19	2998	15810	3	999	0.998
classic	7094	41681	4	1774	0.323
k1b	2340	21819	6	390	0.043
hitech	2301	10800	6	384	0.192
reviews	4069	18483	5	814	0.098
sports	8580	14870	7	1226	0.036
la1	3204	31472	6	534	0.290
la12	6279	31472	6	1047	0.282
la2	3075	31472	6	513	0.274
tr11	414	6424	9	46	0.046
tr23	204	5831	6	34	0.066
tr41	878	7453	10	88	0.037
tr45	690	8261	10	69	0.088

Table 2. Datasets description

5.1 Basic Experiments

We present in this Section an experiment in which all the features of each dataset are used, constructing the graphs as described in Section 4. We first used *dominant set* to extract the prototypical elements of each cluster and then we applied our approach to cluster the remaining data points.

The results of this series of experiments are presented in Table 3. They can be used as point of comparison for our next experiments, in which we used different settings. From the analysis of the table it is not possible to find a stable pattern. The results range from NMI .27 on the *hitech*, to NMI .67 on *k1b*. The reason of this instability is due to the representation of the datasets that in some cases is not appropriate to describe the data.

An example of the graphical representation of the two datasets mentioned above is presented in Fig. 2, where the similarity matrices constructed for *k1b* and *hitech* are shown. We can see that the representation of *hitech* does not show a clear structure near the main diagonal, to the contrary, it is possible to recognize a block structures on the graphs representing *k1b*.

	NG17-19	classic	k1b	hitech	review	sports	la1	la12	la2	tr11	tr23	tr41	tr45
AC	.56 ± .0	.66 ± .07	.82 ± .0	.44 ± .0	.81 ± .0	.69 ± .0	.49 ± .04	.57 ± .02	.54 ± .0	.68 ± .02	.44 ± .01	.64 ± .07	.64 ± .02
NMI	.42 ± .0	.56 ± .22	.66 ± .0	.27 ± .0	.59 ± .0	.62 ± .0	.45 ± .04	.46 ± .01	.46 ± .01	.63 ± .02	.38 ± .0	.53 ± .06	.59 ± .01

Table 3. Results as AC and NMI, with the entire feature space.

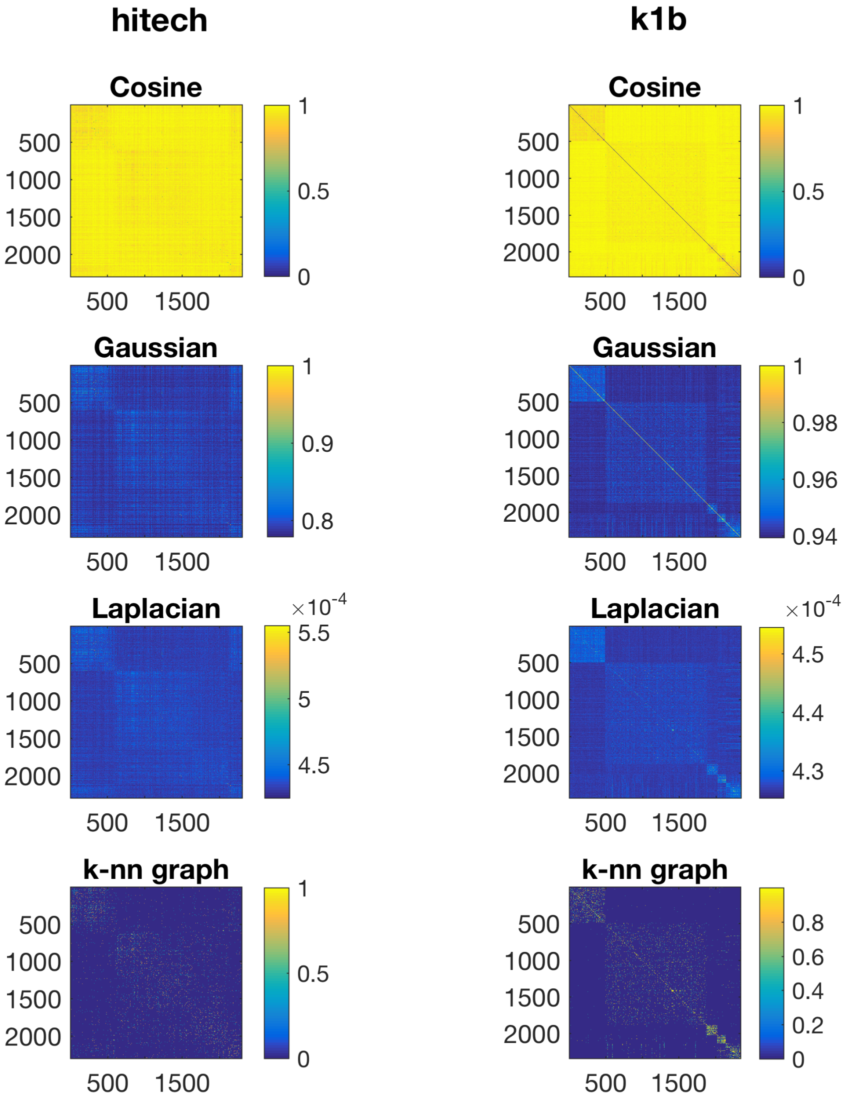

Fig. 2. Different representations for the datasets *hitech* and *k1b*.

5.2 Experiments with Feature Selection

In this section we present an experiment in which we conducted feature selection to see if it is possible to reduce the noise introduced by determined features. To do this, we decided to apply to the corpora a basic frequency selection heuristic that eliminates the features that occur more (or less) often than a determined thresholds. In this study were kept only the words that occur more than once.

This basic reduction leads to a more compact feature space, which is easier to handle. Words that appear very few times in the corpus can be special characters or miss-spelled words and for this reason can be eliminated. The number of features of the reduced datasets are shown in Table 5.2. From the table, we can see that the reduction is significant for 5 of the datasets used, with a reduction of 82% for *classic*. The datasets that are not listed in the table were not affected by this process.

In Table 5.2 we present the results obtained employing feature selection. This technique can be considered a good choice to reduce the size of the datasets and the computational cost, but in this case does not seem to have a big impact on the performances of the algorithm. In fact, the improvements in the performance of the algorithm are not substantial. There is an improvement of 1%, in terms of NMI, in four datasets over five and in one case we obtained lower results. This could be due to the fact that we do not know exactly what features have been removed, because this information is not provided with the datasets. It is possible that the reduction has removed some important (discriminative) word, compromising the representation of the data and the computation of the similarities. Also for this reason we did not use any other frequency selection technique.

	classic	k1b	la1	la12	la2
pre	41681	21819	31472	31472	31472
post	7616	10411	13195	17741	12432
%	0.82	0.52	0.58	0.44	0.6

Table 4. Number of features for each dataset before and after feature selection.

	classic	k1b	la1	la12	la2
AC	.67 ± .0	.79 ± .0	.56 ± .11	.56 ± .03	.57 ± .0
NMI	.57 ± .0	.67 ± .0	.47 ± .12	.44 ± .01	.47 ± .0

Table 5. Mean results as AC and NMI, with frequency selection.

5.3 Experiments with LSA

In this Section we used LSA (see Section 4.1) to reduce the number of features that describe the data. The evaluation was conducted using different numbers of features to describe each dataset, ranging from 10 to 400. This operation is required because there is no agreement on the correct number of features to extract for a determined dataset, for this reason this value has to be calculate experimentally.

The results of this evaluation are shown in two different tables, Table 6 indicates the results as NMI and Table 7 indicates the results as AC for each dataset and number of features. The performances of the algorithm measured as NMI are similar on average (excluding the case of n_v with 10 features), but there is no agreement on different datasets. In fact, different data representations affect heavily the performances on datasets such as *NG17-19*, where the performances ranges from .27 to .46. This phenomenon is due to the fact that each dataset has different characteristics, as shown in Table 2 and that their representation require an appropriate semantic space. With $n_v = 250$ we obtained the higher results on average, both in terms of NMI and AC.

The results with the representation provided by LSA show how this technique is effective in terms of performances. In fact, it is possible to achieve higher results than using the entire feature space or with the frequency selection technique. The improvements are substantial and in many cases are 10% higher. Furthermore, with this new representation it is easier to handle the data.

Data\n_v	10	50	100	150	200	250	300	350	400
NG17-19	.27	.37	**.46**	.26	.35	.37	.36	.37	.37
classic	.53	.63	.71	.73	**.76**	.74	.72	.72	.69
k1b	**.68**	.61	.58	.62	.63	.63	.62	.61	.62
hitech	**.29**	.28	.25	.26	.28	.27	.27	.26	.26
reviews	**.60**	.59	.59	.59	.59	.59	.58	.58	.58
sports	.62	.63	**.69**	.67	.66	.66	.66	.64	.62
la1	.49	.53	.58	.58	.58	.57	**.59**	.57	**.59**
la12	.48	.52	.52	.52	.53	**.56**	.54	.55	.54
la2	.53	.56	.58	.58	.58	.58	**.59**	.58	.58
tr11	.69	.65	.67	.68	**.71**	.70	.70	.69	.70
tr23	.42	**.48**	.41	.39	.41	.40	.41	.40	.41
tr41	.65	.75	.72	.69	.71	.74	**.76**	.69	.75
tr45	.65	**.70**	.67	.69	.69	.68	.68	.67	.69
avg.	.53	.56	**.57**	.56	**.57**	**.57**	**.57**	.56	**.57**

Table 6. NMI results for all the datasets. Each column indicates the results obtained with a reduced version of the feature space using LSA.

Data\n_v	10	50	100	150	200	250	300	350	400
NG17-19	.61	**.63**	.56	.57	.51	.51	.51	.51	.51
classic	.64	.76	.87	.88	**.91**	.88	.85	.84	.80
k1b	.72	.55	.58	.73	**.75**	**.75**	.73	.70	.73
hitech	**.48**	.36	.42	.41	.47	.46	.41	.43	.42
reviews	**.73**	.72	.69	.69	.69	.71	.71	.71	.71
sports	.62	.61	**.71**	.69	.68	.68	.68	.68	.61
la1	.59	.64	.72	.70	**.73**	.72	**.73**	.72	**.73**
la12	.63	.63	.62	.62	.63	**.67**	.64	**.67**	.65
la2	**.69**	.66	.60	.60	.61	.60	.65	.60	.60
tr11	.69	.66	.69	.70	**.72**	.71	.71	.71	.71
tr23	.44	**.51**	.43	.42	.43	.43	.43	.43	.43
tr41	.60	.76	.68	.68	.65	.75	**.77**	.67	**.77**
tr45	.57	**.69**	.66	.68	.67	.67	.67	.67	.67
avg.	.62	.63	.63	.64	.65	**.66**	.65	.64	.64

Table 7. AC results for all the datasets. Each column indicates the results obtained with a reduced version of the feature space using LSA.

5.4 Comparison with State-of-the-art algorithms

The results of the evaluation of the document clustering games are shown in Table 8 and 9 (third column, DCG). We compared the best results obtained with the document clustering games approach and the best results indicated in [38] and in [20]. In the first article it was conducted an extensive evaluation of different generative and discriminative models, specifically tailored for document clustering and two graph-based approaches, CLUTO and a bipartite spectral co-clustering method. In this evaluation the results are reported as NMI and graphical approaches obtained better performances than generative. In the second article were evaluated different NMF approaches to document clustering, on the same datasets, here the results are reported as AC.

From Table 8 it is possible to see that the results of the document clustering games are higher than those of state-of-the-art algorithms on ten datasets out of thirteen. On the remaining three datasets we obtained the same results on two datasets and a lower result in one. On *classic*, *tr23* and tr26 the improvement of our approach is substantial, with results higher than 5%. Form Table 9 we can see that our approach performs substantially better that NMF on all the datasets.

Data	DCG_{noK}	*DCG*	*Best*
NG17-19	.39 ± .0	**.46** ± .0	**.46** ± .01
classic	.71 ± .0	**.76** ± .0	.71 ± .06
k1b	**.73** ± .02	.68 ± .02	.67 ± .04
hitech	**.35** ± .01	.29 ± .02	.33 ± .01
reviews	.57 ± .01	**.60** ± .01	.56 ± .09
sports	.67 ± .0	**.69** ± .0	.67 ± .01
la1	.53 ± .0	**.59** ± .0	.58 ± .02
la12	.52 ± .0	**.56** ± .0	**.56** ± .01
la2	.53 ± .0	**.59** ± .0	.56 ± .01
tr11	**.72** ± .0	.71 ± .0	.68 ± .02
tr23	**.57** ± .02	.48 ± .03	.43 ± .02
tr41	.70 ± .01	**.76** ± .06	.69 ± .02
tr45	**.70** ± .02	**.70** ± .03	.68 ± .05

Table 8. Results as NMI of generative models and graph partitioning algorithm (*Best*) compared to our approach with and without k.

Data	DCG_{noK}	*DCG*	*Best*
NG17-19	.59 ± .0	**.63** ± .0	-
classic	.80 ± .0	**.91** ± .0	.59 ± .07
k1b	**.86** ± .02	.75 ± .03	.79 ± .0
hitech	**.52** ± .01	.48 ± .02	.48 ± .04
reviews	.64 ± .01	**.73** ± .01	.69 ± .07
sports	**.78** ± .0	.71 ± .0	.50 ± .07
la1	.63 ± .0	**.73** ± .0	.66 ± .0
la12	.59 ± .0	**.67** ± .0	-
la2	.55 ± .0	**.69** ± .0	.53 ± .0
tr11	**.74** ± .0	.72 ± .0	.53 ± .05
tr23	**.52** ± .02	.51 ± .05	.43 ± .06
tr41	.75 ± .01	**.77** ± .08	.53 ± .06
tr45	**.71** ± .01	.69 ± .04	.54 ± .06

Table 9. Results as AC of nonnegative matrix factorization algorithms (*Best*) compared to our approach with and without k.

5.5 Experiments with no Cluster Number

In this section we present the experiments conducted with our system in a context in which the number of clusters to extract from the dataset is not used. It has been tested the ability of *dominant set* to find natural clusters and the performances that can be obtained in this context by the document clustering games. We first run *dominant set* to discover many small clusters, setting the parameter of the gaussian kernel with a small value ($\sigma = 0.1$), then these clusters are re-clustered as described in Section 4.4 constructing a graph that encodes their pairwise similarity (see equation 16).

The evaluation of this model was conducted on the same datasets used in previous experiments and the results are shown in Table 8 and 9 (second column, DCG_{noK}). From these tables we can see that this new formulation of the clustering games performs well in many datasets. In fact, in datasets such as *k1b*, *hitech*, *tr11* and *tr23* it has results higher than those obtained in previous experiments. This can be explained by the fact that with this formulation the number of clustered points used by our framework is higher that in the previous experiments. Furthermore, this new technique is able to extract clusters of any shape. In fact, as we can see in Fig. 3, datasets such as *la1* and *la2* present a more compact cluster structure, whereas in datasets such as *k1b* and *hitech* the clusters structure is loose[5].

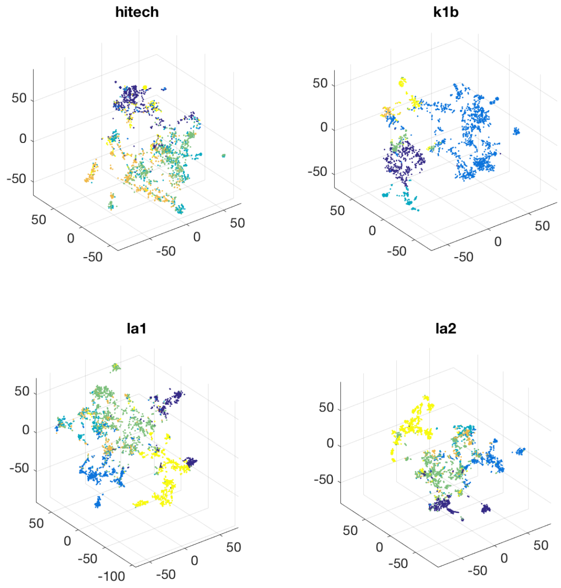

Fig. 3. Representation of the datasets *hitech*, *k1b*, *la1* and *la2*.

The performances of the system can be improved with this setting when it is able to extract the exact number of natural clusters from the graph. To the contrary, when it is not able to predict this number, the performances decrease drastically. This phenomenon can explain why this approach performs poorly in some datasets. In fact, in datasets such as, *NG17-19*, *la1*, *la12* and *l2* the system performs poorly compared to our previous experiments. In many cases this happens because during the clustering phase we extract more clusters than expected. The results as NMI of our system are higher than those of related algorithms on 8 over 13 datasets, even if k is not given as input. Also the results as AC are good, in fact on 9 datasets over 11 we obtained better performances.

[5] The dataset have been visualized using t-SNE to reduce the features to 3d.

5.6 Experiments on streaming data

In this section we present the evaluation of our approach on streaming datasets. For this task we used the same datasets used in previous experiments but this time we divided each of them in 12 random folds. In this way we simulated the data streaming process, clustering the data iterativelly. We performed the experiments 15 times to not bias our test sets. For each experiment we selected a random fold as initial clustering and performed 11 runs of the algorithm, each time including a new fold in the test set. Previous clusterings are used to drive the choices of new data points to specific clusters, making the final clustering coherent.

Data	2	3	4	5	6	7	8	9	10	11	12
ng17-19	.57 ± .07	.55 ± .05	.55 ± .04	.55 ± .03	.55 ± .03	.55 ± .03	.55 ± .03	.55 ± .03	.55 ± .03	.55 ± .03	.55 ± .03
classic	.81 ± .02	.81 ± .02	.81 ± .02	.81 ± .01	.81 ± .01	.81 ± .01	.81 ± .01	.81 ± .01	.81 ± .01	.81 ± .01	.81 ± .01
k1b	.85 ± .04	.83 ± .03	.83 ± .02	.83 ± .02	.83 ± .02	.83 ± .01	.83 ± .01	.83 ± .01	.83 ± .02	.83 ± .01	.83 ± .01
hitech	.38 ± .04	.34 ± .04	.34 ± .03	.33 ± .03	.33 ± .02	.32 ± .02	.32 ± .02	.32 ± .02	.32 ± .02	.32 ± .02	.32 ± .02
reviews	.77 ± .03	.75 ± .02	.75 ± .02	.74 ± .02	.74 ± .01	.74 ± .02	.74 ± .02	.74 ± .01	.74 ± .02	.74 ± .01	.74 ± .01
sports	.86 ± .02	.85 ± .02	.84 ± .02	.84 ± .01	.84 ± .01	.83 ± .01	.83 ± .01	.83 ± .01	.83 ± .01	.83 ± .01	.83 ± .01
la1	.65 ± .05	.63 ± .04	.63 ± .04	.63 ± .03	.64 ± .02	.64 ± .02	.63 ± .02	.63 ± .02	.63 ± .02	.63 ± .02	.63 ± .02
la12	.68 ± .03	.67 ± .02	.66 ± .01	.67 ± .01	.66 ± .01	.66 ± .01	.66 ± .01	.66 ± .01	.66 ± .01	.66 ± .01	.66 ± .01
la2	.68 ± .03	.67 ± .02	.67 ± .02	.67 ± .02	.66 ± .02	.66 ± .01	.66 ± .01	.67 ± .01	.67 ± .01	.67 ± .01	.67 ± .02
tr11	.69 ± 10	.64 ± .09	.61 ± 10	.58 ± .08	.56 ± .08	.56 ± .07	.55 ± .07	.54 ± .07	.54 ± .07	.54 ± .07	.54 ± .07
tr23	.66 ± 11	.57 ± 10	.52 ± .08	.50 ± .09	.50 ± .08	.49 ± .08	.48 ± .09	.48 ± .09	.47 ± .08	.46 ± .08	.45 ± .08
tr41	.86 ± .05	.84 ± .05	.83 ± .04	.83 ± .04	.83 ± .03	.82 ± .03	.82 ± .03	.82 ± .03	.82 ± .03	.82 ± .03	.81 ± .03
tr45	.79 ± .04	.76 ± .04	.76 ± .04	.75 ± .04	.74 ± .04	.74 ± .04	.73 ± .04	.73 ± .03	.73 ± .03	.73 ± .04	.73 ± .04

Table 10. Results as NMI for all the datasets. Each column indicates the results obtained including the corresponding fold in the test set.

Data	2	3	4	5	6	7	8	9	10	11	12
ng17-19	.85 ± .03	.84 ± .03	.84 ± .02	.84 ± .01	.84 ± .02	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01
classic	.94 ± .01	.94 ± .01	.94 ± .01	.94 ± .01	.94 ± .00	.94 ± .00	.94 ± .00	.94 ± .00	.94 ± .00	.94 ± .00	.94 ± .00
k1b	.94 ± .02	.94 ± .01	.94 ± .01	.94 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.94 ± .01	.94 ± .01	.94 ± .01
hitech	.61 ± .04	.61 ± .03	.61 ± .03	.61 ± .03	.60 ± .02	.60 ± .02	.60 ± .02	.60 ± .02	.60 ± .02	.60 ± .02	.60 ± .02
reviews	.92 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01	.91 ± .01
sports	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01	.95 ± .01
la1	.82 ± .03	.82 ± .03	.82 ± .02	.82 ± .02	.82 ± .02	.82 ± .02	.82 ± .02	.82 ± .02	.82 ± .01	.82 ± .01	.82 ± .01
la12	.85 ± .02	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .00	.84 ± .01	.84 ± .00	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .00
la2	.83 ± .02	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01	.84 ± .01
tr11	.72 ± .07	.72 ± .08	.71 ± .08	.70 ± .07	.69 ± .07	.69 ± .06	.69 ± .06	.69 ± .06	.69 ± .06	.69 ± .06	.69 ± .06
tr23	.73 ± .08	.71 ± .08	.69 ± .08	.69 ± .07	.69 ± .07	.68 ± .07	.68 ± .07	.68 ± .07	.68 ± .07	.68 ± .07	.68 ± .07
tr41	.90 ± .04	.90 ± .03	.90 ± .03	.90 ± .03	.90 ± .02	.90 ± .02	.90 ± .02	.90 ± .02	.90 ± .02	.90 ± .02	.90 ± .02
tr45	.80 ± .04	.81 ± .04	.82 ± .04	.82 ± .04	.82 ± .04	.82 ± .03	.82 ± .04	.82 ± .03	.82 ± .03	.82 ± .04	.82 ± .04

Table 11. Results as AC for all the datasets. Each column indicates the results obtained including the corresponding fold in the test set.

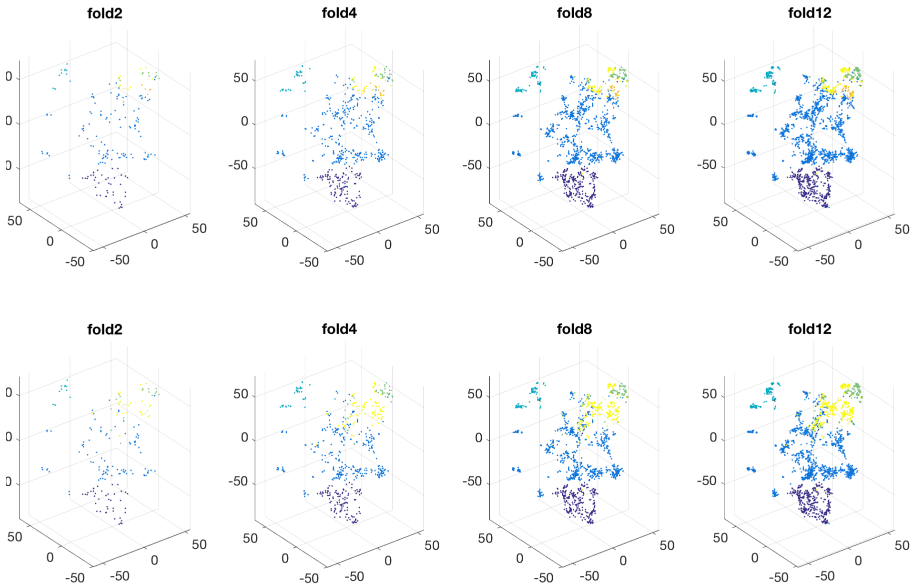

Fig. 4. Visualizations of the results on *k1b* on different folds. The upper row shows the ground truth and the lower row shows the results of our approach.

The results of this evaluation are presented in Table 10 and 11 as NMI and AC, respectively. From the tables we can see that the performances of the system are stable over time. In fact, we can see that in 9 datasets over 13, the different among the results as NMI with the entire dataset (12 folds) and those with 2 folds is 2%. The results as AC are even better. In fact, with the entire dataset the performances are stable and in two cases higher (*la2* and *tr45*). The latter behavior can be explained considering the fact that the algorithm exploit contextual information and in many cases having more information to use leads to better solutions. We can see that just in one case we have a drop of 5% in performances, comparing the results in fold 2 with those in fold 12. The most negative results have been achieved on small datasets, this because in these cases the clusters are small and unbalanced. In particular dealing with clusters of very different sizes makes the *k*-nn algorithm, used to sparsify the graph, not useful. In fact, the resulting structure allow the elements of small clusters to have connections with elements belonging to other clusters. In these cases the dynamics of our system converge to solutions in which small clusters are absorbed by bigger ones. This because the elements belonging to small clusters are likely to receive influence from the elements belonging to large clusters if *k* is larger than the cardinality of the small clusters. This phenomenon can be seen in Fig. 4, where we compare the clustering results of our method against the ground truth, on *k1b*. We can see that the orange cluster disappears in fold

2 and that this error is propagated on the other folds. The other clusters are partitioned correctly.

If we compare the results in this Section with the results proposed in Section 5.4 we can see that with this approach we can have a bust in performances. In fact, in all datasets, except one (tr11) the results are higher both in terms of NMI and AC. We can see that using just few labeled points allows our approach to substantially improve its performances. Furthermore we see that these performance are stable over time and that the standard deviation is very low in all experiments, ≤ 0.11 for NMI and ≤ 0.8 for AC.

Comparison with k-nn We conducted the same experiment described in previous Section to compare the performances of our method with the k-nearest neighbor (k-NN) algorithm. We used k-NN to classify iteratively the folds of each dataset treating the data in same way of previous experiments and setting $k = 1$. Experimentally we noticed that this value achieve the best performances. Higher values have very low NMI, leading to situations in which small clusters are merged in bigger ones.

Data	2	3	4	5	6	7	8	9	10	11	12
ng3sim	.25 ± .03	.31 ± .03	.36 ± .03	.40 ± .02	.43 ± .02	.46 ± .01	.48 ± .01	.49 ± .01	.51 ± .01	.52 ± .01	.53 ± .01
classic	.31 ± .02	.39 ± .02	.44 ± .02	.49 ± .02	.52 ± .01	.55 ± .01	.58 ± .01	.60 ± .01	.62 ± .01	.63 ± .01	.64 ± .01
k1b	.32 ± .04	.38 ± .03	.44 ± .02	.49 ± .02	.53 ± .02	.57 ± .02	.60 ± .01	.62 ± .01	.64 ± .01	.66 ± .01	.67 ± .01
hitech	.17 ± .03	.18 ± .02	.20 ± .02	.21 ± .02	.23 ± .02	.24 ± .01	.26 ± .01	.27 ± .01	.28 ± .01	.29 ± .01	.29 ± .01
reviews	.35 ± .03	.41 ± .03	.46 ± .02	.50 ± .02	.53 ± .02	.55 ± .01	.57 ± .01	.59 ± .01	.60 ± .01	.61 ± .01	.62 ± .01
sports	.48 ± .02	.56 ± .02	.62 ± .01	.66 ± .01	.69 ± .01	.71 ± .01	.73 ± .01	.75 ± .01	.76 ± .01	.77 ± .00	.78 ± .00
la1	.31 ± .02	.35 ± .02	.39 ± .02	.42 ± .02	.44 ± .02	.46 ± .02	.48 ± .01	.50 ± .01	.51 ± .01	.52 ± .01	.53 ± .01
la12	.32 ± .02	.37 ± .02	.41 ± .01	.45 ± .01	.48 ± .01	.50 ± .01	.52 ± .01	.53 ± .01	.55 ± .01	.56 ± .01	.57 ± .01
la2	.33 ± .03	.37 ± .03	.41 ± .02	.44 ± .02	.47 ± .01	.49 ± .01	.51 ± .01	.53 ± .01	.54 ± .01	.55 ± .01	.56 ± .01
tr11	.36 ± .07	.38 ± .04	.40 ± .04	.43 ± .04	.45 ± .03	.47 ± .03	.49 ± .03	.50 ± .02	.52 ± .02	.53 ± .02	.54 ± .02
tr23	.34 ± .12	.35 ± .09	.39 ± .06	.40 ± .06	.41 ± .07	.44 ± .06	.46 ± .06	.47 ± .05	.49 ± .04	.50 ± .04	.52 ± .04
tr41	.41 ± .05	.47 ± .04	.51 ± .03	.55 ± .03	.59 ± .02	.61 ± .02	.63 ± .02	.65 ± .02	.67 ± .02	.68 ± .02	.70 ± .01
tr45	.46 ± .05	.48 ± .05	.52 ± .04	.55 ± .03	.57 ± .02	.60 ± .02	.62 ± .02	.63 ± .02	.64 ± .02	.65 ± .01	.66 ± .01

Table 12. Results as NMI for all the datasets using k-NN. Each column indicates the results obtained including the corresponding fold in the test set.

The results of this evaluation are shown in Table 12 and 13 as NMI and AC, respectivelly. From these tables we can see that the performances of k-NN are not stable and tend to increase at each step. We can notice that the results in fold 2 in many cases are doubled in fold 12, this behaviour demonstrate that this algorithm requires many data to achieve good classification performances. To the contrary with our approach it is possible to obtain stable performances in each fold.

The performances of k-NN are very low compared with our approaches. In particular, we can see that it does not perform well in the first seven folds. This can be explained considering that it classify new instances taking into account only local information (the information on the class membership of its nearest

Data	2	3	4	5	6	7	8	9	10	11	12
ng3sim	.60 ± .02	.67 ± .02	.72 ± .01	.76 ± .01	.78 ± .01	.80 ± .01	.81 ± .01	.82 ± .01	.83 ± .01	.84 ± .01	.84 ± .00
classic	.59 ± .02	.68 ± .01	.73 ± .01	.77 ± .01	.80 ± .01	.82 ± .01	.84 ± .01	.85 ± .00	.86 ± .00	.87 ± .00	.87 ± .00
k1b	.53 ± .04	.62 ± .02	.69 ± .02	.74 ± .01	.78 ± .01	.81 ± .01	.83 ± .01	.84 ± .01	.86 ± .01	.87 ± .01	.88 ± .01
hitech	.40 ± .03	.44 ± .02	.48 ± .02	.51 ± .02	.53 ± .01	.55 ± .01	.57 ± .01	.58 ± .01	.59 ± .01	.60 ± .01	.61 ± .01
reviews	.58 ± .03	.66 ± .02	.72 ± .01	.76 ± .01	.78 ± .01	.81 ± .01	.82 ± .01	.83 ± .00	.84 ± .00	.85 ± .00	.86 ± .00
sports	.72 ± .01	.79 ± .01	.83 ± .01	.86 ± .01	.88 ± .00	.89 ± .00	.90 ± .00	.91 ± .00	.92 ± .00	.92 ± .00	.93 ± .00
la1	.46 ± .02	.55 ± .01	.61 ± .01	.66 ± .01	.69 ± .01	.71 ± .01	.73 ± .01	.74 ± .01	.76 ± .01	.77 ± .01	.78 ± .01
la12	.49 ± .01	.58 ± .01	.64 ± .01	.68 ± .01	.72 ± .01	.74 ± .01	.76 ± .01	.77 ± .01	.78 ± .01	.79 ± .00	.80 ± .00
la2	.49 ± .03	.58 ± .02	.64 ± .02	.68 ± .01	.71 ± .01	.73 ± .01	.75 ± .01	.76 ± .01	.78 ± .01	.78 ± .00	.79 ± .00
tr11	.42 ± .05	.43 ± .04	.46 ± .04	.50 ± .04	.55 ± .03	.58 ± .03	.61 ± .02	.63 ± .02	.66 ± .02	.67 ± .02	.69 ± .02
tr23	.49 ± .07	.49 ± .05	.54 ± .04	.59 ± .04	.63 ± .04	.66 ± .04	.69 ± .04	.71 ± .03	.73 ± .03	.75 ± .03	.76 ± .03
tr41	.45 ± .05	.50 ± .03	.55 ± .02	.62 ± .02	.67 ± .02	.71 ± .01	.73 ± .01	.76 ± .01	.78 ± .01	.80 ± .01	.81 ± .01
tr45	.50 ± .04	.56 ± .04	.62 ± .03	.67 ± .02	.71 ± .02	.74 ± .01	.76 ± .01	.78 ± .01	.79 ± .01	.80 ± .01	.81 ± .01

Table 13. Results as AC for all the datasets using k-NN. Each column indicates the results obtained including the corresponding fold in the test set.

neighbour), without considering any other source of information and without imposing any coherence constraint using contextual information.

Form Table 12 and 13 we can see that the results of k-NN in fold 12 (entire dataset) are almost always lower that those obtained with our method, both in terms of NMI and AC. In fact, just in two cases k-NN obtain equal and higher results, in *tr11* and *tr23* if we consider the NMI. If we consider the results as AC we can see that in two datasets k-NN has the same performances of our method (*NG17-19* and *tr11*) and that it has higher performances on *hitech* (+1%).

6 Conclusions

With this work we explored new methods for document clustering based on game theory and consistent labeling principles. We have conducted an extensive series of experiments to test the approach on different scenarios. We have also evaluated the system with different implementations and compared the results with state-of-the-art algorithms.

Our method can be considered as a continuation of graph based approaches but it combines together the partition of the graph and the propagation of the information across the network. With this method we used the structural information about the graph and then we employed evolutionary dynamics to find the best labeling of the data points. The application of a game theoretic framework is able to exploit relational and contextual information and guarantees that the final labeling of the data is consistent.

The system has demonstrated to perform well compared with state-of-the-art system and to be extremely flexible. In fact, it has been tested with different features, with and without the information about the number of clusters to extract and on static and dynamic context. Furthermore, it is not difficult to implement new graph similarity measure and new dynamics to improve its performances or to adapt to new contexts.

The experiments without the use of K, where the algorithm collects together highly similar points and then merges the resulting groups, demonstrated how it is able to extract clusters of any size without the definition of any centroid. The experiments on streaming data demonstrated that our approach can be used to cluster data dynamically. In fact, the performances of the system does not change much when the test set is enriched with new instances to cluster. This is an appealing feature, since it makes the framework flexible and not computationally expensive. On this scenario it was demonstrated that the use of contextual information helps the clustering task. In fact, using the k-NN algorithm on streaming data produces lower and not stable results.

As future work we are planning to apply this framework to other kind of data and also to use it in the context of *big data*, where, in many cases, it is necessary to deal with datasets that do not fit in memory and have to be divided in different parts in order to be clustered or classified.

Acknowledgments

This work was partly supported by the Samsung Global Research Outreach Program.

References

1. Aggarwal, C.C.: A survey of stream clustering algorithms. In: Data Clustering: Algorithms and Applications, pp. 231–258. IEEE (2013)
2. Aggarwal, C.C.: Data Streams: Models and Algorithms. Springer (2014)
3. Ardanuy, M.C., Sporleder, C.: Structure-based clustering of novels. EACL 2014 pp. 31–39 (2014)
4. Bharat, K., Curtiss, M., Schmitt, M.: Methods and apparatus for clustering news content (Jul 28 2009), uS Patent 7,568,148
5. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (Mar 2003), `http://dl.acm.org/citation.cfm?id=944919.944937`
6. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 269–274. ACM (2001)
7. Ding, C., Li, T., Peng, W.: Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence chi-square statistic, and a hybrid method. In: Proceedings of the national conference on artificial intelligence. vol. 21, p. 342. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 (2006)
8. Erdem, A., Pelillo, M.: Graph transduction as a noncooperative game. Neural Computation 24(3), 700–723 (2012)
9. Haykin, S., Network, N.: A comprehensive foundation. Neural Networks 2(2004) (2004)
10. Jain, A.K., Dubes, R.C.: Algorithms for clustering data. Prentice-Hall, Inc. (1988)
11. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse processes 25(2-3), 259–284 (1998)
12. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401(6755), 788–791 (1999)

13. Leyton-Brown, K., Shoham, Y.: Essentials of game theory: A concise multidisciplinary introduction. Synthesis Lectures on Artificial Intelligence and Machine Learning 2(1), 1–88 (2008)
14. Lovasz, L.: Matching theory (north-holland mathematics studies). Annals of Discrete Mathematics (1986)
15. Manning, C.D., Raghavan, P., Schütze, H., et al.: Introduction to information retrieval, vol. 1. Cambridge university press Cambridge (2008)
16. Nowak, M.A., Sigmund, K.: Evolutionary dynamics of biological games. science 303(5659), 793–799 (2004)
17. Okasha, S., Binmore, K.: Evolution and rationality: decisions, co-operation and strategic behaviour. Cambridge University Press (2012)
18. Pavan, M., Pelillo, M.: Dominant sets and pairwise clustering. Pattern Analysis and Machine Intelligence, IEEE Transactions on 29(1), 167–172 (2007)
19. Peterson, A.D.: A Separability Index for Clustering and Classification Problems with Applications to Cluster Merging and Systematic Evaluation of Clustering Algorithms. Ph.D. thesis, Iowa State University (2011)
20. Pompili, F., Gillis, N., Absil, P.A., Glineur, F.: Two algorithms for orthogonal nonnegative matrix factorization with application to clustering. Neurocomputing 141, 15–25 (2014)
21. Rota Bulò, S., Pelillo, M.: A game-theoretic approach to hypergraph clustering. IEEE transactions on pattern analysis and machine intelligence 35(6), 1312–1327 (2013)
22. Rota Buló, S., Pelillo, M., Bomze, I.M.: Graph-based quadratic optimization: A fast evolutionary approach. Computer Vision and Image Understanding 115(7), 984–995 (2011)
23. Sandholm, W.H.: Population games and evolutionary dynamics. MIT press (2010)
24. Sankaranarayanan, J., Samet, H., Teitler, B.E., Lieberman, M.D., Sperling, J.: Twitterstand: news in tweets. In: Proceedings of the 17th acm sigspatial international conference on advances in geographic information systems. pp. 42–51. ACM (2009)
25. Shawe-Taylor, J., Cristianini, N.: Kernel methods for pattern analysis. Cambridge university press (2004)
26. Smith, J.M., Price, G.: The logic of animal conflict. Nature 246, 15 (1973)
27. Strehl, A., Ghosh, J.: Cluster ensembles – a knowledge reuse framework for combining multiple partitions. The Journal of Machine Learning Research 3, 583–617 (2003)
28. Szabó, G., Fath, G.: Evolutionary games on graphs. Physics Reports 446(4), 97–216 (2007)
29. Tagarelli, A., Karypis, G.: Document clustering: The next frontier. Data Clustering: Algorithms and Applications p. 305 (2013)
30. Taylor, P.D., Jonker, L.B.: Evolutionary stable strategies and game dynamics. Mathematical biosciences 40(1), 145–156 (1978)
31. Tripodi, R., Pelillo, M.: A game-theoretic approach to word sense disambiguation. Computational Linguistics (in press)
32. Tripodi, R., Pelillo, M.: Document clustering games. In: Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods. pp. 109–118 (2016)
33. Von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behavior (60th Anniversary Commemorative Edition). Princeton University Press (1944)
34. Weibull, J.W.: Evolutionary game theory. MIT press (1997)

35. Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. pp. 267–273. ACM (2003)
36. Zhao, Y., Karypis, G.: Empirical and theoretical comparisons of selected criterion functions for document clustering. Machine Learning 55(3), 311–331 (2004)
37. Zhao, Y., Karypis, G., Fayyad, U.: Hierarchical clustering algorithms for document datasets. Data mining and knowledge discovery 10(2), 141–168 (2005)
38. Zhong, S., Ghosh, J.: Generative model-based document clustering: a comparative study. Knowledge and Information Systems 8(3), 374–384 (2005)