Replicator Graph Clustering

Michael Donoser
http://vh.icg.tugraz.at

Institute for Computer Graphics and Vision, Graz University of Technology, Austria

Abstract

In this paper we introduce an efficient, effective and scalable clustering method denoted as *Replicator Graph Clustering*. Our method takes measures of similarity between pairs of data points (i. e. an affinity matrix) as input and identifies a set of clusters and unique cluster assignments in a fully unsupervised manner, where the cluster granularity is adaptable by a single parameter. We provide clustering results in three subsequent steps: (a) diffusing affinities by finding personalized evolutionary stable strategies of non-cooperative games (b) building a mutual k-nearest neighbor graph representing the underlying manifold and (c) applying a graph based clustering strategy which identifies the final clusters. Individual steps have low computational complexity which leads to an efficient clustering method, scaling well with an increasing number of data points. Experimental evaluation demonstrates competitive performance to state-of-the-art in several application fields.

1 Introduction

In this paper we address the generic problem of data clustering, which aims at identifying a varying number of clusters and unique cluster assignments in a fully unsupervised manner. Standard approaches like k-means or the mean shift [5] are defined for Euclidean metrics. To overcome the shortcomings of this Euclidean assumption, researchers started to consider the underlying similarity manifold to be able to handle non-metric data in the field of clustering [25, 26]. Such manifold approaches were also considered in several other interrelated application fields like retrieval, semi-supervised learning, non-linear dimensionality reduction and metric learning. Since our proposed clustering method is highly related to these manifold approaches, we first give an overview of the related work in these fields.

Retrieval aims at identifying and ranking the most similar elements to a provided query from a potentially huge database. In this field, manifold analysis is frequently used to improve the retrieval performance by re-evaluating the similarities between elements in the context of the entire database. The most popular concept for this re-evaluation is based on diffusing the similarities considering the underlying manifold based on the random walk principle. The random walk moves between elements of the database analyzing a transition matrix that defines probabilities for walking from one element to another one. The probabilities are fixed proportional to the provided affinities, yielding higher probabilities for making steps between more similar elements. By repeatedly making random walk steps, similarities are diffused on the manifold, which in turn improves the obtainable retrieval scores. Such diffusion processes were *e.g*. proposed in [2, 30, 31, 32] and in a recent paper [6] these methods are summarized in a general diffusion framework.

Retrieval is highly related to semi-supervised learning methods, since retrieval can be seen as an extreme case of semi-supervised learning, where the only labeled instance is the query element. In this field the most popular methods are graph based, as e. g. semi-supervised learning by label propagation [34], Gaussian random field models and harmonic functions [35] and local and global consistency analysis [33].

Also the field of dimensionality reduction frequently addresses the non-linearity of data by manifold analysis. Nonlinear reduction techniques like Isomap [28], Locally Linear Embedding [23] or Laplacian eigenmaps [3] are frequently applied for non-linear embedding, mapping the input data to different graph structures by defining a neighborhood for each data-point in terms of a graph structure.

In metric learning, the insight is considered that the default distance in a feature space may not be optimal. Methods in this field aim at detecting the manifold structure and additionally the associated metric. Unsupervised examples are *e.g*. the probabilistic latent semantic analysis [14] (pLSA) and the latent dirichlet allocation (LDA) [4].

Also state-of-the-art clustering methods like spectral clustering [18], power iteration clustering [15] or affinity propagation [11] are highly related to the aforementioned topics. In spectral clustering the top eigenvectors of the graph Laplacian unfold the affinity manifold to form meaningful clusters. The cluster results are then obtained by traditional methods like k-means in the newly obtained space. Methods in this fields mainly differ in how to normalize the Graph Laplacian [10, 19, 24]. Power iteration clustering focuses on efficiency issues of spectral approaches, and shows that by directly analyzing the principal eigenvector of the affinity matrix, one is able to derive reasonable clusters in short computation time. All these approaches require to manually fix the number of clusters. By contrast, in affinity propagation [11] messages (responsibilities and availabilities) are exchanged between the data points until a high-quality set of exemplars and corresponding clusters emerge and in such a way the number of clusters is implicitly identified. Cluster granularity is adaptable by a so-called preference parameter.

Our goal is to combine ideas and approaches from the aforementioned fields of research and to propose a novel unsupervised clustering method which provides results in an efficient manner. The main idea is that we first apply a diffusion process considering the underlying data manifold to overcome the shortcomings of a Euclidean assumption as *e.g*. used in standard k-means clustering and to then use the diffused affinities in a provably optimal clustering approach which analyzes an effective neighborhood graph structure (*Replicator Graph*). Our proposed clustering method has several advantages:

- No restrictions (like affinities have to satisfy metric properties) on affinity matrix, e. g. negative and asymmetric affinity matrices can be handled.
- Diffusion step improves pairwise affinities due to considering the underlying manifold.
- Returns global optimal clustering solution considering widespread cluster criterium of high internal coherency and external incoherency.
- Automatic identification of number of clusters using a single parameter to influence cluster granularity.
- Improved clustering quality and reduced computation time in comparison to state-of-the-art methods.

2 Replicator Graph Clustering

Our clustering method, denoted as *Replicator Graph Clustering (RPC)*, belongs to the field of pairwise or proximity-based based clustering approaches, which assume that the input is an $N \times N$ affinity matrix $\mathbf{A} = (a_{ij})$, where each entry a_{ij} measures the similarity between two specific data points-to-be-clustered i and j, and we do not have a self-similarity, i. e. a_{ii} is 0. Please note, that this is quite different to methods centered around the notion of features such as the standard k-means method. In our setting we have lost knowledge of the underlying representation (features) and only have access to the provided similarities. The goal of clustering is to uniquely assign each of the N data points to one of a set of clusters $\mathcal{C} = (\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_C)$, where C is an automatically found number of clusters. In such a way the clustering result depends exclusively on the provided similarities.

Our method is divided into three subsequent steps. First, we diffuse the provided affinities through the entire matrix considering the underlying manifold in a game theoretical setting. For this, we describe in Section 2.1 an approach based on personalized evolutionary stable strategies that represent optimally propagated affinities per data point. The updated affinity matrix is then used to define a mutual k-nearest neighbor graph, as it is described in Section 2.2. Finally, this graph representation is passed to a provably optimal clustering algorithm, which identifies clusters by analyzing internal and external cluster consistency scores. This clustering step is described in Section 2.3.

2.1 Diffusing Affinities by Game Theory

The first step of our method is to diffuse the provided affinities considering the underlying data manifold. In such a way we lift the Euclidean assumption prevalent in many other clustering approaches. It has been shown in [6] that such diffusion processes are able to significantly improve retrieval performance, and we aim at exploiting this property for our task of clustering. Our main idea is to consider each data point independently and to diffuse the affinities in relation to the current query data point, similar as in related semi-supervised learning or retrieval tasks. Query-specific diffusion means, that we aim at converting the query specific similarities $A_{i.}$ (the i-th row of the given affinity matrix $\mathbf{A}$) into a new $N \times 1$ vector representation $\mathbf{x}^*$ by maximizing its agreement to the underlying similarity manifold spanned by the entire affinity matrix $\mathbf{A}$ by

$$\mathbf{x}^* = \underset{\mathbf{x}}{argmax} \left(\mathbf{x}^T \mathbf{A}\, \mathbf{x}\right) \quad , \tag{1}$$

which is a standard quadratic assignment problem (QAP). Any of the available QAP solvers can be applied, and we describe replicator dynamics as an effective solver for this problem in Section 2.1.1. The QAP is known to be NP-hard, and thus all solvers will return an arbitrary local maximum and it is in general not possible to guarantee that the query point is in the basin of attraction of the obtained solution. Thus, in Section 2.1.2 we introduce an adaption for personalizing the replicator dynamics to a specific data point i, which returns a query specific affinity vector $\mathbf{x}^{i*}$ containing optimally propagated affinities for the query i. Finally, all query-specific diffused affinities stacked together build the updated affinity matrix $\mathbf{A}^*$, which considers the similarity distribution on the underlying manifold.

2.1.1 Replicator Dynamics

Replicator dynamics [27] is a popular algorithm from game theory, which is frequently applied for solving the quadratic assignment problem. Game theory in general is a tool for predicting how players behave in strategic situations (games). Each player has a set of available actions and the obtainable reward depends on the actions played by each player. In this paper we focus on non-cooperative, two player games, where players are in a competitive setting. Final result of the game is an *evolutionary stable strategy* which is the outcome of the evolutionary process that unfolds over time finding the best strategy. Game theoretical approaches were recently gaining increased popularity in computer vision, e. g. for matching image segments and points [1], for finding common spatial visual patterns in images [17], for semi-supervised learning [7] but also for clustering [16, 22].

The dynamics are an iterative procedure, starting from a random initialization $\mathbf{x}^0$, which is iteratively updated by

$$x_i^{t+1} = x_i^t \, \frac{(\mathbf{A}\,\mathbf{x}^t)_i}{{\mathbf{x}^t}^T \mathbf{A}\,\mathbf{x}^t} \ , \tag{2}$$

where $\mathbf{x}^t$ is the assignment vector at time t. As a necessary additional constraint $\mathbf{x}$ has to lie on the simplex Δ defined as

$$\Delta = \left\{ \mathbf{x} \in \mathbb{R}^N : x_i \geq 0 \quad and \quad \mathbf{1}^T \mathbf{x} = 1 \right\} \ , \tag{3}$$

where $\mathbf{1}$ is an N-dimensional vector of ones, i. e. $\sum x_i = 1$. Of course, also the random initialization vector $\mathbf{x}_0$ has to lie on the simplex, and mostly a slightly perturbed version of the barycenter of the simplex is used. Starting from $\mathbf{x}^0$, replicator dynamics find a solution vector $\mathbf{x}^*$ which is a local (!) maximum of the optimization problem shown in Equation 1 (QAP problem).

The simplex Δ is invariant under this formulation, which means that every trajectory starting on the simplex will remain on the simplex. Furthermore, the score defined in Equation 1 is strictly increasing along any trajectory of the dynamics given in Equation 2. The final solution is the *evolutionary stable strategy*, which is a stricter formulation of the well-known Nash equilibrium. As it is shown in [29] the dynamics also converge in case of negative and asymmetric affinity matrices by considering a shifted matrix $\mathbf{B} = \mathbf{A}^T + \alpha\mathbf{I}$, with $0.5 < \alpha < 1$ (guarantees unambiguous solutions) and $\mathbf{I}$ is the identity matrix. Please note, that in such a way we do not have any restrictions on the affinity matrix, i. e. it can be negative and asymmetric, which is quite different from most affinity based clustering approaches like [11, 19, 24]. For more details and convergence proofs of these dynamics see e. g. [20].

A nice property of the replicator dynamics algorithm is that it can be implemented in a few lines of code in any programming language (Matlab Pseudo Code shown in Algorithm 1).

Algorithm 1: Replicator Dynamics Code

```
random initialization of $\mathbf{x}$
while $dist \leq \varepsilon$ do
    $\mathbf{old_x} = \mathbf{x}$
    $\mathbf{x} = \mathbf{x}.*(\mathbf{A}*\mathbf{x})$
    $\mathbf{x} = \mathbf{x}./sum(\mathbf{x})$
    $dist = ||\mathbf{x} - \mathbf{old_x}||$
```

The intuition behind considering replicator dynamics for our diffusion process is as follows: the hypotheses, that the data points contribute to the score defined in Equation 1, compete with each other. Each data point gains support from compatible points and competitive pressure from all other data points during the evolutionary process. This competitive setting reduces the number of non-zero assignments by driving inconsistent hypotheses to extinction (i. e. making $\mathbf{x}^*$ sparse) and by increasing the support of important data points. In such a way affinities defined in $\mathbf{A}$ are diffused through the manifold.

2.1.2 Personalized Replicator Dynamics

Replicator dynamics only guarantee to find an arbitrary local maxima $\mathbf{x}^*$. Since our goal is to use the dynamics to diffuse the affinities for a specific query data point i, we are interested in finding the local maxima that has data point i in its basin of attraction, which is not guaranteed if applying the standard dynamics using random initialization. For this reason, instead of initializing the dynamics by a slightly perturbed version of the barycenter of the simplex, we consider the query-specific affinities $\mathbf{A}_{i.}$ (the i-th row of the given affinity matrix $\mathbf{A}$) as initialization. Since, for replicator dynamics the starting vector $\mathbf{x}^0$ has to lie on the simplex, we have to normalize it by

$$\overline{\mathbf{A}}_{i.} = \frac{\mathbf{A}_{i.}}{\sum \mathbf{A}_{i.}}, \tag{4}$$

which ensures that $\overline{\mathbf{A}}_{i.}$ lies on the simplex. The starting vectors $\mathbf{x}^0$ are then fixed to the corresponding $\overline{\mathbf{A}}_{i.}$ vectors.

Additionally, we locally constrain the diffusion process to the S nearest neighbors (NN) of each query point i, and in such a way not all pairs of affinities contribute to the dynamics. For this we have to find the S-NN for each i. We then replace the corresponding row $\mathbf{A}_{i.}$ of the affinity matrix $\mathbf{A}$ with a vector that only has values for the identified neighbors and all other affinities are fixed to zero. This leads to a novel affinity matrix, which we denote as $\mathbf{A}_{SNN}$. Note that this matrix is fixed for all personalized replicator dynamics steps per query, and that these dynamics only differ in their initialization.

We then apply the evolutionary dynamics for each query i using $\mathbf{A}_{SNN}$ and the corresponding $\overline{\mathbf{A}}_{i.}$ as initialization vector $\mathbf{x}^0$ in Equation 2, which yields optimal $\mathbf{x}^{i*}$ per query. In such a way, we are finally able to derive the updated $N \times N$ affinity matrix $\mathbf{A}^*$ as

$$\mathbf{A}^* = \begin{bmatrix} \left[\mathbf{x}^{1*}\right]^T \\ \vdots \\ \left[\mathbf{x}^{N*}\right]^T \end{bmatrix} \tag{5}$$

where each row is the personalized local maxima obtained by the dynamics.

In general, we have to apply the dynamics for each query, and each replicator dynamics diffusion has $O\left(N^2\right)$ memory and time complexity. Therefore in overall we would have a complexity of $O\left(N^3\right)$ for building the updated matrix $\mathbf{A}^*$. Nevertheless, as aforementioned, we only use the S-NN in the diffusion, which reduces complexity from N^3 to NS^2, where $S \ll N$. In fact as it is shown in the experimental section S can be fixed to only a fraction of N (e. g. 3%) resulting in the same clustering quality. Our game theoretical diffusion algorithm is summarized in Algorithm 2. As can be seen, runtime of our algorithm could potentially be significantly decreased by parallelizing the main loop.

Algorithm 2: Algorithm for Game Theoretical Diffusion of Affinities

```
Input: $N \times N$ Affinity matrix $\mathbf{A}$ and number of neighbors $S$
Output: Updated $N \times N$ Affinity matrix $\mathbf{A}^*$
1 $\mathbf{A}^* = []$
2 Calculate S-nearest neighbors per query and build $\mathbf{A}_{SNN}$
3 for $i = 1$ to $N$ do
4     Apply replicator dynamics to $\mathbf{A}_{SNN}$ using $\overline{\mathbf{A}}_{i.}$ as initialization and obtain
      evolutionary stable strategy $\mathbf{x}^{\mathbf{i}*}$ (Equation 2)
5     $\mathbf{A}^* = \begin{bmatrix} \mathbf{A}^* \\ \left[\mathbf{x}^{i*}\right]^T \end{bmatrix}$
6 return $\mathbf{A}^*$
```

2.2 Building the Replicator Graph

Most of the methods presented in the introduction are based on representing the similarity manifold by a graph structure, where nodes represent data points and edges define similarities between the data points. We follow the same principle and use a graph structure (which we denote as *Replicator Graph*) as underlying representation for the final clustering step described in the next section. Obviously it is important to select a good graph representation.

Conventional nearest neighbor graphs are supposed to model the local relation between a data point i and e. g. its k nearest neighbors (kNN) or all points within a pre-defined distances $A(i, j) < \varepsilon$ (εNN). Depending on k or ε, the resulting graph is typically very sparse, compared to a fully connected graph constructed from a dense affinity matrix $\mathbf{A}$. Nevertheless, both graph structures have obvious drawbacks. The use of ε-neighborhood graphs requires data distributions on same scales which means that the data points have to form non-elongated, tight clusters. In contrast, k nearest neighbor graphs are able to connect regions on different scales, since only the absolute neighborhood ranking is of interest for the connection of two points. Nevertheless, a kNN graph fails to consider possible asymmetries in the affinity matrix. Since we potentially have to deal with such asymmetries of $\mathbf{A}^*$, we focus on a specific graph structure denoted as mutual kNN graph ($mkNN$), which is surprisingly seldom used despite its interesting properties. The $mkNN$ adds an additional constraint to the kNN graph, which requires that two connected data points belong to each others k-neighborhood, i. e. $j \in k\mathrm{NN}(i)$ and $i \in k\mathrm{NN}(j)$. In such a way we obtain our *Replicator Graph* by building the $mkNN$ graph structure for the affinity matrix $\mathbf{A}^*$, which represents the underlying manifold in a sparse manner, but contains all relevant affinities for our final clustering step as it is described in the next section. The computational complexity for building the replicator graph is $O\left(k^2 N\right)$, where finding the set of kNN requires sorting the matrix which is possible in linear time for $k \ll N$.

2.3 Replicator Graph Clustering

The replicator graph, as described in the previous section, represents the underlying affinity manifold and constitutes the basis for our global optimal graph clustering. We propose to cluster the obtained replicator graph according to an unsupervised segmentation method introduced by Felzenszwalb and Huttenlocher [9]. In [9] a highly efficient segmentation method was introduced, which uses relative dissimilarities between segments to determine plausible merges. It is formalized as a grouping problem on a graph, where nodes represent image pixels, edges are defined e. g. by standard 4-neighborhood and edge weights are obtained by measuring local color similarities. Authors introduced an algorithm that provably optimizes a global grouping metric and efficiently provides unsupervised segmentation results in $O(n \log n)$ time, where n is the number of image pixels.

We adapt [9] to general graph structures for our unsupervised, proximity based clustering method. We assume that we have given the *Replicator Graph* as described in the previous section, where nodes represent the individual data points that we want to cluster and edges connect nodes according to the *mkNN* graph with edge weights set according to the updated affinity matrix $\mathbf{A}^*$. We use this graph as input to [9], which precedes by merging clusters in decreasing order of the edge weights separating them. This is efficiently done by a variant of a minimum spanning tree method. These iterative merging steps stop once the final clustering is neither too fine (cluster should be merged due to high similarity) nor too coarse (cluster should be split due to boundary evidence). In such a way the algorithm obeys the global properties of being neither too fine nor too coarse and returns a global optimal(!) solution, although the algorithm only makes greedy decisions. For more details on the algorithm and proofs of the global optimality, see [9]. The computational complexity of this approach is $O(E \log E)$, where E is the number of edges in the graph which is directly related to number of neighbors k considered in our replicator graph.

An important part of the algorithm is the so-called goodness function $\tau(C_i)$ of a cluster C_i, which is fixed to $\tau(C_i) = f/|C_i|$, where f is a region size preference parameters. Setting the parameter f defines the desired granularity of the clustering and therefore directly influences the number of clusters in a way similar to the preference parameter of affinity propagation clustering [11]. Nevertheless it is important to note, that the preference parameter f does not (!) represent a minimum cluster size. Smaller clusters are allowed, if there is a sufficiently large difference between the clusters.

3 Experiments

Experimental evaluation is split into three main parts. In Section 3.1 we evaluate the influence of the main parameters of our method. In Section 3.2 we demonstrate the applicability of the game theoretical diffusion step for clustering. Finally, Section 3.3 compares results to related methods on diverse data sets. We implemented our method in Matlab and code is provided at `http://vh.icg.tugraz.at`.

3.1 Parameter Evaluation

This section analyzes the influence of the parameters on clustering performance. Experiments are based on a popular data set from the field of shape retrieval: MPEG-7 CE Shape-1 Part-B dataset, which consists of 70 different classes of object silhouettes, where each class is represented by 20 instances, thus $N = 1400$. In order to define the 1400×1400 affinity matrix A, we use the currently best performing shape matching method [12] on MPEG-7. For evaluating the performance of a clustering method we use the wide-spread normalized mutual information (NMI) score.

We have three main parameters to define for our clustering method: the fraction size S for locally constraining the diffusion (influencing speed), the neighborhood size k for the *mkNN* graph (influencing speed), and the clustering granularity parameter f (influencing the

[%] of N	0.2	0.6	0.9	1.4	1.8	2.1	2.5	3.6	50.0
S	3	8	12	20	25	30	35	50	700
NMI · 100	76.89	91.62	95.13	96.70	97.18	97.35	97.46	97.32	97.36
time [sec]	0.99	0.81	0.90	1.31	1.46	1.68	1.98	3.04	559.05

Table 1: Analysis of influence of parameter S on the clustering quality (normalized mutual information – NMI – the higher the better) and the runtime. Reasonable clustering performance is already achieved by only considering a fraction of N in the diffusion ($S \sim 35$ nearest neighbors) yielding a significant speedup of factor 500.

k	2	3	4	5	7	10	15	20	25
NMI · 100	93.45	96.51	96.01	97.39	97.46	97.46	97.46	97.46	97.46
time [sec]	1.86	1.87	1.89	1.90	1.93	1.98	2.04	2.10	2.14

Table 2: Analysis of influence of nearest neighbor parameter k on the clustering quality (normalized mutual information – NMI – the higher the better). Optimal clustering performance is achieved by considering values of $k > 7$, for higher values performance stays exactly the same, but at increased runtime, since the complexity of clustering is $O(E \log E)$, where E is the number of edges (directly related to k).

number of clusters). Table 1 analyzes the influence of the parameter S (number of nearest neighbors to consider in the dynamics) on the obtainable clustering quality and algorithm runtime if k is fixed to 10. As can be seen only a fraction of data points (3%) has to be considered in the dynamics to achieve reasonable performance but within a significantly decreased runtime. Increasing the neighborhood size S further slightly decreases performance, and selecting an optimal size is still open research in diffusion, see e. g. [21]. The neighborhood size k does not influence the quality of the results as long as a sufficiently large number such as $k > 7$ is considered as it is shown in Table 2 (for $S = 35$). Adapting k also has only a minor influence on runtime.

3.2 Evaluation of Diffusion

As second experiment we demonstrate the usefulness of the game theoretical diffusion step for clustering. Figure 1 shows the normalized mutual information (NMI) score for MPEG-7 comparing clustering results obtained by activating the replicator dynamics based diffusion of affinities and results if directly using the input matrix (i. e. skipping the first step of our method). As can be seen the diffusion step provides improved results independent of the obtained number of clusters, where different cluster results were obtained by adapting the granularity parameter f. Additionally, we show scores if replacing our diffusion scheme by two related methods: Label Propagation (LP) [31] and LDCP [2], which are both outperformed by our approach.

3.3 Comparison to State-of-the-art

As final experiment we compare our clustering method on several data sets to related methods in the field of clustering. We use a k-means baseline, spectral clustering [24], affinity propagation [11] and the recently proposed, highly efficient power iteration clustering [15]

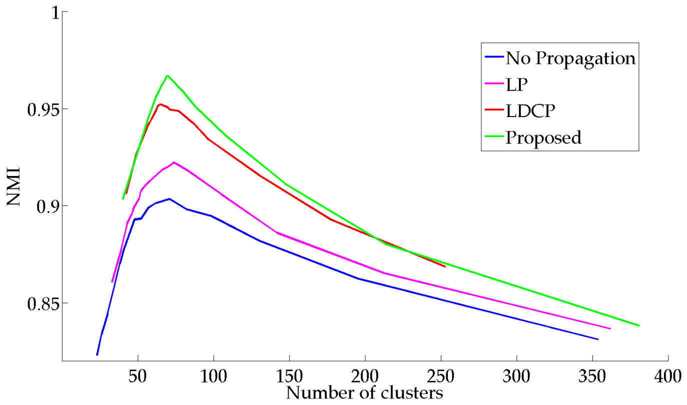

Figure 1: Illustration of improved performance when using the proposed replicator dynamics based diffusion scheme. Activating the diffusion scheme leads to improved performance independent of the number of clusters obtained.

Data	K-means	Spectral [24]	Aff. Prop. [11]	PIC [15]	Proposed
MPEG	89.19 (**1.05s**)	95.99 (3.45s)	91.11 (13.72s)	47.08 (1.38s)	**97.46** (2.69s)
CALT	41.19 (2.15s)	55.75 (1.33s)	55.65 (5.99s)	51.01 (1.43s)	**56.40** (**0.73s**)
UNIP	42.59 (**0.01s**)	68.49 (0.04s)	59.58 (0.54s)	64.88 (0.03s)	**71.66** (0.18s)

Table 3: Comparison of clustering quality (normalized mutual information – the higher the better) and runtime (in seconds) for all methods compared on several data sets.

for comparison. Code was taken from the corresponding author's webpages. Since spectral clustering and power iteration clustering apply k-means on the obtained embedding as final step, we passed the correct number of clusters as additional parameter for these methods, whereas for affinity propagation and our method, the preference value f was adapted to always yield exactly (!) the same number of clusters.

We analyze three different data sets: (1) MPEG-7 CE Shape-1 Part-B (shape silhouettes, 1400 instances, 70 classes) as in the previous section, (2) CALTECH 101 (object categories, 1010 instances, 101 clusters) [8] and (3) UNIPEN (handwritten letters, 250 instances, 5 classes) [13]. We fixed the parameters for our method to $S = 30$ and $K = 10$. Table 3 shows results of the Normalized Mutual Information (NMI) score and runtimes on these data sets for all of the 5 compared methods. As can be seen our proposed method provides competitive clustering results on all data sets in short computation time, e. g. being significantly faster than affinity propagation [11]. In contrast, the power iteration clustering is quite fast, nevertheless fails if the number of classes gets too high, since it finally applies k-means to a simple one-dimensional vector. Huge improvements against the k-means baseline (+8.27% on MPEG-7, +15.21% on CALTECH 101 and +29.07% on UNIPEN) demonstrate the importance of considering the underlying manifold for clustering, which is implicitly done by the diffusion scheme.

4 Conclusion

In this paper we proposed a fully unsupervised, proximity-based clustering method denoted as *Replicator Graph Clustering*. Our method combines an effective diffusion process, based on iteratively approaching evolutionary stable strategies, with a provably optimal clustering step. We furthermore described an approach to significantly decrease computation time by considering only nearest neighbors in the dynamics, which nevertheless does not adversely affect clustering quality. In such a way effective clustering results are obtained in low computation time. Experiments first demonstrate that the diffusion scheme improves clustering due to considering the underlying data manifold and that competitive results to related clustering methods are obtainable on diverse clustering data sets.

References

[1] A. Albarelli, S. Rota Bulo, A. Torsello, and M. Pelillo. Matching as non-cooperative game. In *Proc. Intern. Conf. on Computer Vision (ICCV)*, 2009.

[2] X. Bai, X. Yang, L.J. Latecki, W. Liu, and Z. Tu. Learning context-sensitive shape similarity by graph transduction. *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 32:861–874, 2010.

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[5] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 24, 2002.

[6] M. Donoser and H. Bischof. Diffusion processes for retrieval revisited. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[7] A. Erdem and M. Pelillo. Graph transduction as a noncooperative game. *Neural Computation*, 24(3):700–723, 2012.

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *CVPR Workshop on generative-Model Based Vision*, 2004.

[9] P. Felzenszwalb and F. Huttenlocher. Efficient graph-based image segmentation. *Intern. Journal of Computer Vision*, 59(2):167–181, 2004.

[10] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nystrĺom method. *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 26:214–225, 2004.

[11] B.J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[12] R. Gopalan, P. Turaga, and R. Chellappa. Articulation-invariant representation of non-planar shapes. In *Proc. European Conf. on Computer Vision (ECCV)*, 2010.

[13] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. Unipen project of on-line data exchange and recognizer benchmarks. In *Proc. Intern. Conf. on Pattern Recognition (ICPR)*, 1994.

[14] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence*, 1999.

[15] F. Lin and W.W. Cohen. Power iteration clustering. In *Proc. Intern. Conf. on Machine Learning (ICML)*, 2010.

[16] H. Liu and S. Yan. Robust graph mode seeking by graph shift. In *Proc. Intern. Conf. on Machine Learning (ICML)*, 2010.

[17] H. Liu and S. Yan. Common visual pattern discovery via spatially coherent correspondences. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[18] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007. ISSN 0960-3174.

[19] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[20] M. Pelillo. Matching free trees with replicator equations. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[21] V. Premachandran and R. Kakarala. Consensus of k-nns for robust neighborhood selection on graph-based manifolds. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[22] S. Rota Bulo, A. Torsello, and M. Pelillo. A game theoretic approach to partial clique enumeration. *Image and Vision Computing*, 27:911–922, 2009.

[23] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[24] J. Shi and J. Malik. Normalized cuts and image segmentation. *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 22:888–905, 2000.

[25] R. Souvenir and R. Pless. Manifold clustering. In *Proc. Intern. Conf. on Computer Vision (ICCV)*, 2005.

[26] R. Subbarao and P. Meer. Nonlinear mean shift for clustering over analytic manifolds. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1168–1175, 2006.

[27] P. Taylor and Jonker L.B. Evolutionarily stable strategies and game dynamics. *Mathematical Biosciences*, 40, 1978.

[28] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[29] A. Torsello, S. Rota Bulo, and M. Pelillo. Grouping with asymmetric affinities: A game-theoretic perspective. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 292–299, 2006.

[30] B. Wang and Z. Tu. Affinity learning via self-diffusion for image segmentation and clustering. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] X. Yang, S. Koknar-Tezel, and L.J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[32] X. Yang, L. Prasad, and L.J. Latecki. Affinity learning with diffusion on tensor product graph. *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 35, 2013.

[33] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *Advances in Neural Information Processing Systems (NIPS)*. 2004.

[34] X. Zhu. *Semi-Supervised Learning with Graphs*. PhD thesis, Language Technologies Institute, Carnegie Mellon University, 2005.

[35] X. Zhu, Z. Ghahramani, and J.D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. Intern. Conf. on Machine Learning (ICML)*, pages 912–919, 2003.