ITGC: Information-theoretic Grid-based Clustering

Sahar Behzadi
Faculty of Computer Science
University of Vienna
Vienna, Austria
sahar.behzadi@univie.ac.at

Hermann Hinterhauser
Faculty of Computer Science
University of Vienna
Vienna, Austria
a00946522@unet.univie.ac.at

Claudia Plant
Faculty of Computer Science and ds:Univie
University of Vienna
Vienna, Austria
claudia.plant@univie.ac.at

ABSTRACT

Grid-based clustering algorithms are well-known due to their efficiency in terms of the fast processing time. On the other hand, when dealing with arbitrary shaped data sets, density-based methods are most of the time the best options. Accordingly, a combination of grid and density-based methods, where the advantages of both approaches are achievable, sounds interesting. However, most of the algorithms in these categories require a set of parameters to be specified while usually it is not trivial to appropriately set them. Thus, we propose an **I**nformation-**T**heoretic **G**rid-based **C**lustering (ITGC) algorithm by regarding the clustering as a data compression problem. That is, we merge the neighbour grid cells (clusters) when it pays off in terms of the compression cost. Our extensive synthetic and real-world experiments show the advantages of ITGC compared to the well-known clustering algorithms.

1 INTRODUCTION

Among various clustering approaches some of them attract more attentions because of their advantages. Partition-based clustering algorithms are popular due to their simplicity and the relative efficiency [7], [2]. K-means [7] is a well-know and well-studied representative for this approach where initially the data is partitioned into k non-empty sets and iteratively the data points are assigned to their nearest cluster. Despite the mentioned advantages, the clustering algorithms in this group suffer from some drawbacks. For instance, the number of clusters k should be specified in the beginning and the results are not deterministic because of their sensitivity to the initialization. Moreover, they are not suitable to discover clusters with non-convex shapes. As a subset of this group, model-based clustering algorithms consider a specific distribution model to represent the data sets. Among them, Expectation-Maximization (EM) algorithm interpret the data as a mixture of Gaussian distributions [5]. On the other hand, density-based clustering algorithms [6], [3] are appropriately designed to deal with clusters having an arbitrary shape. Unlike the partition-based algorithms, the algorithms in this approach are able to deal with noisy data sets. However, in order to find dense regions we need to specify two parameters representing the radius and the density of a neighborhood. Additionally, density-based algorithms are not designed to efficiently deal with clusters with various densities. Spectral clustering [9] is another approach which has become popular due to its simple implementation and its performance in many graph-based clustering. It can be solved efficiently by any standard linear algebra software. However, this approach is expensive for the large data sets since the Computing eigenvectors is the bottleneck.

Another well-known approach is grid-based clustering where any data set is partitioned using a set of grid-cells and data points are assigned to the appropriate grid cell. Grid-based methods [1], [14], [11] quantize the object space into a finite number of cells (hyper-rectangles) and then perform the required operations on the quantized space. The main advantage of grid-based methods is their fast processing time which depends on the number of cells in the grid. In the other word, no distance computation is required and the clustering is performed on summaries and not on the individual objects. Thus, the complexity of grid-based algorithms is usually *O(number of populated grid cells)* and not *O(number of objects)*. Beyond their ability to deal with noisy data sets, grid-based clustering algorithms are able to identity clusters irrespective of their shapes. Unlike most of the clustering algorithm which require an initialization phase, the algorithms in this category are insensitive to the order of input records and therefore are deterministic.

Despite the valuable advantages of grid-based clustering algorithms, to the best of our knowledge, all of them are parametric algorithms where a user is required to specify the parameters. However, most of the time it is not trivial to appropriately set them. Thus, utilizing the principle of Minimum Description Length (MDL) we propose a non-parametric **I**nformation-**T**heoretic **G**rid-based **C**lustering algorithm where we regard the clustering task as a data compression problem so that the best clustering is linked to the strongest data compression. First, an adaptive grid is constructed corresponding to the statistical characteristics of any data set and non-empty cells are considered as single clusters. Then, we combine the concept of density and grid-based methods and employing our compression-based objective function we start merging clusters with their neighbour grid cells only if it pays off in terms of the compression cost.

In this paper we propose an information-theoretic clustering algorithm offering the following contributions:

- **Adaptive partitioning:** We utilize the statistical characteristics of any data set, e.g. local and global dispersion, in order to introduce an adaptive partitioning of the data.
- **Non-parametric clustering:** Employing the MDL-based objective function, we iteratively merge clusters when it pays off in terms of the compression cost automatically. Thus, no parameter needs to be specified.
- **Insensitivity to the shape of clusters:** ITGC employs the concept of density-based methods in order to select the next merging candidate. Thus, it is insensitive to the shape of clusters whether they are Gaussian, arbitrary or even having various density regions.
- **Scalability:** Analogous to other grid-based clustering algorithm, the complexity of ITGC depends on the number of cells not on the number of objects which leads to a scalable algorithm in terms of the number of objects.

2 INFORMATION-THEORETIC GRID-BASED CLUSTERING

In order to introduce a grid-based clustering algorithm we need to address two fundamental questions: (1) how to find a specific appropriate partitioning (grid) corresponding to any data set; (2) how to efficiently merge the cells to discover the hidden clusters. Thus, our proposed algorithm ITGC consists of two main building blocks: (1) finding a suitable grid corresponding to specific characteristics of any data set and (2) employing MDL principle to effectively and efficiently merge the cells without any parameter to be specified.

2.1 Partitioning the Data

Finding a suitable partitioning with respect to the data is a crucial task in a grid-based clustering algorithm. Inspired by [8], we utilize the characteristics of any data set to introduce the best fitting partition. That is, we are looking for a partition which leads to high internal homogeneity in the cells and high external heterogeneity of each cell with respect to its neighbors for every single cell. Thus, for any cell C_j consisting of n_j data points the statistical indicators are defined as:

$$\bar{X_j} = \frac{\sum_{i=1}^{n_j} X_{ij}}{n_j} \quad and \quad S_j = \sqrt{\frac{\sum_{i=1}^{n_j}(X_{ij} - \bar{X_j})^2}{n_j - 1}} \tag{1}$$

Where X_{ij} is the distance of the $i - th$ data point in C_j to the center of this cell. Thus, $\bar{X_j}$ is the average distance of data points to the center of C_j and S_j is the standard deviation of the cell. These are statistical indicators on the local level (each individual cell), similar indicators are calculated on the global level (the entire grid) as:

$$\mu = \frac{\sum_j \bar{X_j}}{N} \quad and \quad \sigma = \sqrt{\frac{\sum_j(\bar{X_j} - \mu)^2}{N - 1}} \tag{2}$$

Where μ is the average center - distance of all cells, N is the total number of cells and σ is the standard deviation of all cells. Based on these indicators, we define $CV_{Local}(j)$ and CV_{Global} as the coefficient of variation (CV) corresponding to any cell C_j and the global variation, respectively. That is,

$$CV_{Local(j)} = \frac{S_j}{\bar{X_j}} \quad and \quad CV_{Global} = \frac{\sigma}{\mu} \tag{3}$$

In another point of view, the above scores show how wide-spread the data points are indicating the relative dispersion at the local (cell) and global (grid) levels. Finally the partitioning cost is defined as:

$$gridCost = \frac{CV_{Global}}{avg \quad CV_{Local}(j)} \tag{4}$$

Considering the cost corresponding to any grid size $k \times k$, we iteratively increase k starting from 1 until it pays off. However, it is not trivial to justify a terminal for this process without observing its trend. Initially, the grid cost increases sharply by increasing k then it slows down quickly and continues linearly. Observing this common trend, we conduct a simple linear regression on various costs with respect to various k values. The regression line is expected to fit more through the higher ks where the costs have lower deviations. Thus, the optimal partitioning can be set to the first k where the grid cost deviated from the fitted line lower than the average.

On the other side, by increasing the size of grid the area of non-empty cells decreases. Thus, it is reasonable to assume this trend to continue with even smaller cells, but the descending trend slows down while decreasing cell sizes. Visualizations of the collected area reveals a common trend which is reverse to the previous one. The area starts at a maximum value, decreases very sharply at lower k and keeps decreasing at a lower gradient. In order to find the optimum value for k, we analogously fit a linear regression through the data set rejecting the low k values which deviate larger than average from the fitted line. The following steps summarize this procedure.

- Step 1: The grid is divided into $k \times k$ cells where the initial size for k is 1.
- Step 2: The grid cost as well as the area of non-empty cells are determined and the values are stored.
- Step 3: We iteratively increase k ranging from 1 to a max_k and repeat the previous steps
- Step 4: Now the optimum partitioning is determined employing two different criteria.

2.2 MDL-based Objective Function

Utilizing the Minimum Description Length (MDL) principle [10] we regard the clustering task as a data compression problem so that the best clustering is linked to the strongest data compression. Given the appropriate model corresponding to any attribute, MDL leads to an intuitive clustering result employing the compression cost as a clustering criterion. The better the model matches major characteristics of the data, the better the result is. Following the MDL principle, we encode not only the data but also the model itself and minimize the overall description length. Simultaneously, we avoid over-fitting since the MDL tends to a natural trade-off between model complexity and the goodness-of-fit. That is, for a given cluster C_i the corresponding compression cost is defined as:

$$MDL(C_i) = CodingCost(C_i) + ParamCost(C_i) + IDCost(C_i) \tag{5}$$

where *CodingCost* shows the cost of coding the data points in cluster C_i by means of a coding scheme. The next two terms illustrate the model complexity where the model itself needs to be encoded. In this paper we employ the Huffman coding scheme to encode the data considering an appropriate model. That is, given the corresponding *Probability Distribution Function* (PDF) to any attribute, the coding cost of any object x is determined by $-log_2 PDF(x)$. Any PDF would be applicable and using a specific model is not a restriction [4] for our algorithm. In this paper, we consider Gaussian PDF for simplicity. In the following we elaborate our objective function more concretely.

- **Objective Function:** The overall MDL-based objective function is the summation of the all compression costs with respect to various clusters. That is,

$$MDL(\mathcal{D}) = \sum_{C_i \in C} MDL(C_i) \tag{6}$$

  where $\mathcal{D}$ is the entire data set and $C = \{C_1, ..., C_k\}$ is the set of all clusters.

- **Data Coding Cost:** Let $\mathcal{X} = \{X_1, ..., X_d\}$ denote the set of all attributes. For any object $x = (x_1, ..., x_d)$ the corresponding coding cost is the sum of encoding any attribute value x_i. Putting all together, the coding cost corresponding to cluster C_i is given by:

$$CodingCost(C_i) = -\sum_{X_j \in \mathcal{X}} \sum_{x \in C_i} \log_2 PDF_j(x) \tag{7}$$

where $PDF_j(.)$ is the Gaussian model with respect to $j-th$ attribute X_j.

- **Model Complexity:** Without taking the model complexity into account, the best result will be a clustering consisting of singleton clusters. This result is completely useless in terms of the interpretation. In order to specify the associated cluster with any data object, we need to encode the cluster IDs. Thus, the IDCosts are required to balance the size of clusters and defined as:

$$IDCost(C_i) = |C_i|.log_2 \frac{|C_i|}{|\mathcal{D}|} \tag{8}$$

Following the fundamental results from the information theory [10], for any attribute X_j the parameters corresponding to model employed to encode the data need to be encoded as well. That is, concerning any Gaussian distribution PDF_j with respect to the attribute X_j, the mean value and the standard deviation need to be encoded, i.e.

$$ParamCost(C_i) = \frac{1}{2}.(2|\mathcal{X}|).log_2|C_i| \tag{9}$$

2.3 Algorithm

As mentioned, ITGC consists of two main building blocks. Algorithm 1 summarizes our grid-based algorithm ITGC. First, an optimal grid is constructed following the steps mentioned in Section 2.1, i.e. the procedure *FindOptimumGrid(.)*. Then, we start merging the cells if it pays off in terms of our objective function (Section 2.2). Initially every cell is considered as a cluster while empty cells are ignored. The cluster with the most number of data points is chosen in the sense that at the end the results are deterministic. We compute the coding cost with respect to the selected cluster MDL_{before} and merge this cluster with one of its neighbors and compute the cost after merging two clusters MDL_{after}. If the cost after merging is smaller than the cost before, we merge two clusters and continue the merging process. Otherwise, the visited cluster is marked. Finally the algorithm terminates if no unmarked non-empty cell exists.

3 EXPERIMENTS

In this section we assess the performance of ITGC comparing to other clustering algorithms in terms of *Normalized Mutual Information* (NMI) which is a common evaluation measure for clustering results [13]. NMI numerically evaluates pairwise mutual information between ground truth and resulted clusters scaling between zero and one.

We conducted several experiments evaluating our algorithm on synthetic and real-world data sets. In order to investigate the effectiveness of ITGC we generated various data sets and compared to the base-line clustering algorithms, i.e. k-means [7] and DBSCAN [6]. While the insensitivity of ITGC to the shape of clusters as well as its effectiveness is illustrated by synthetic experiments, we extended the comparison to the wider range of well-known clustering algorithms. Our algorithm is implemented in Java and the source code as well as the data sets are publicly available [1].

[1]https://tinyurl.com/y85gglpx

Algorithm 1: Information-theoretic grid-based clustering

```
ITGC ($\mathcal{D}$)
$\mathcal{G}$ = FindOptimumGrid($\mathcal{D}$);
$C = \{C_1, ..., C_k\}$        // Non-empty cells in $\mathcal{G}$
seeds := non-visited clusters;
while  (seeds != empty) do
    $C_i$ := the cluster with the most data points in $C$
    $C_i$ is visited
    while ($MDL_{before} > MDL_{after}$) do
        $MDL_{before} = MDL(C_i)$
        $C_j$ := a random non-visited neighbor cell w.r.t $C_i$
        $C_m$ := the cluster after merging $C_i$ and $C_j$
        $MDL_{after} = MDL(C_m)$
        if $MDL_{before} > MDL_{after}$ then
            remove $C_j$ and $C_i$ from $C$ .
            add $C_m$ to $C$
        end if
    end while
    seeds := non-visited clusters;
end while
return ($C$)
```

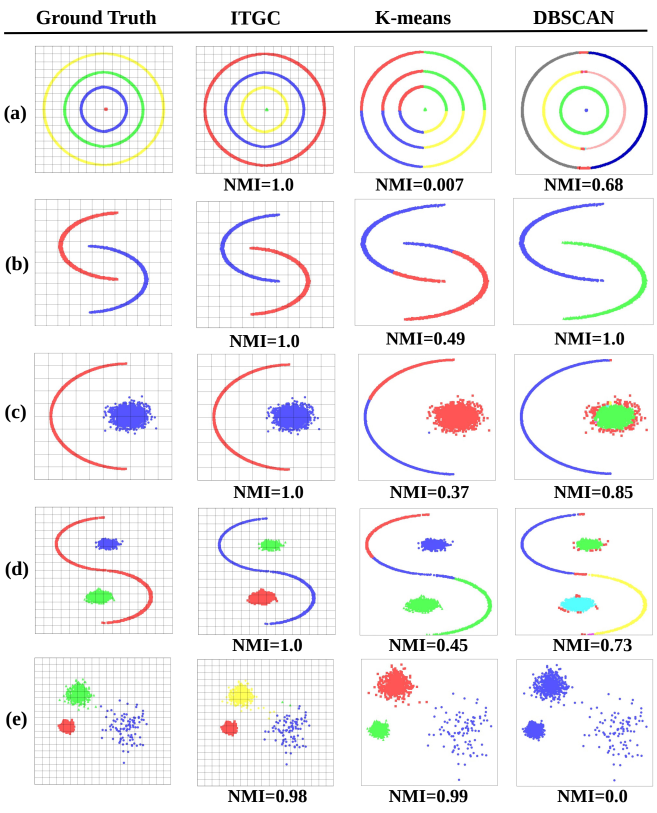

Figure 1: Comparison on various synthetic data sets.

3.1 Synthetic Experiments

In order to cover all aspects of ITGC, we investigate the performance of the algorithms considering various synthetic data sets including arbitrary shaped data sets as well as clusters with different densities. Then, we continue experiments by comparing all algorithms in terms of the scalability.

Performance: Most of the time any clustering algorithm is designed for a specific kind of data sets. For instance, k-means

Dataset	*Attr./Objects*	*ITGC*	*k-means*	*DBSCAN*	*EM*	*Spectral C.*	*CLIQUE*	*Single L.*
Iris	4/150	**0.66**	0.53	0.59	0.60	0.6	0.00	0.59
Occupancy Detection	7/20560	**0.61**	0.56	0.00	0.31	0.00	**0.61**	-
Breast Cancer	9/286	**0.47**	0.32	0.41	0.45	0.45	0.39	0.27
User Knowledge	5/403	0.24	**0.27**	0.01	**0.27**	**0.27**	0.00	0.01

Table 1: Comparison on real data sets.

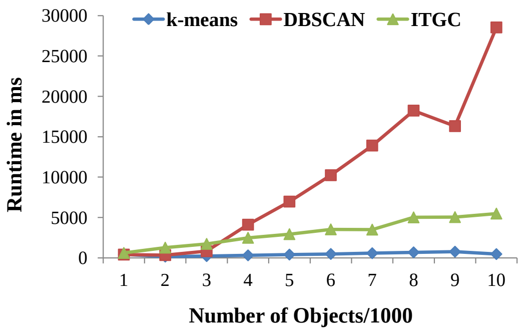

Figure 2: Scalability of the algorithms by increasing the number of objects.

appropriately deals with Gaussian shaped data sets and its performance dramatically decreases when the clusters have no specific shape. On the other hand density-based clustering algorithms are sensitive to different densities w.r.t. various clusters. In order to evaluate the performance of ITGC concerning various shapes of clusters, we synthetically generated arbitrary shaped clusters in a combination with some Gaussian clusters. Figure 1 shows the effectiveness and the insensitivity of ITGC considering various cases. As expected, k-means fails when the clusters are not Gaussian (Figure 1a,b,c,d). On the other hand, DBSCAN is not able to discover the clusters with various densities (Figure 1d,e)

Scalability: To evaluate the efficiency in terms of the runtime complexity we generated 5 dimensional synthetic data sets where we iteratively increased the number of data objects ranging from 1,000 to 10,000. Figure 2 shows the result of this experiment. As expected, k-means is the fastest algorithm while DBSCAN is the worse since its complexity highly depends on the number of objects. Although ITGC is not able to outperform k-means, its corresponding execution time is still reasonable and more efficient than DBSCAN.

3.2 Real Experiments

In this section we extend our experiments to the wider range of clustering algorithms including EM [5], Single link [12], spectral clustering [9] and CLIQUE [1] as the well-known representatives for any clustering approach. We evaluate clustering quality of ITGC on real-world data sets. We used *Iris*, *Occupancy Detection*, *User Knowledge* and *Breast Cancer* data sets from the UCI Repository [2]. Table 1 shows the characteristics of any data set and the results of applying various algorithms in terms of NMI. Concerning any data set the best NMI is high lighted and when getting "Out Of Memory" error we inserted "-" in the table. As Table 1 illustrates ITGC outperforms other algorithms considering the first 3 real-world data sets. Interestingly, in this experiment we outperform CLIQUE which is a well-known grid and density-based clustering algorithm (the results are similar on the *Occupancy* data set). Although some of the comparison methods perform slightly better than ITGC on *User Knowledge* data set, our result is still comparable and we outperform DBSCAN, CLIQUE and Single link.

[2]http://archive.ics.uci.edu/ml/index.php

4 CONCLUSION AND FUTURE WORKS

In this paper we propose an information-theoretic clustering algorithm, ITGC, utilizing the MDL-principle. Firstly, We employ the statistical characteristics of any data set to appropriately partition the data without any presumptions. Then, an MDL-based objective function is proposed to iteratively merge the neighbour clusters when it pays of in terms of the compression cost of the clusters. Our experiments on synthetic and real-world data sets show the advantages of our proposed algorithm compared to other well-known clustering algorithms. Similar to other grid-based clustering algorithms, our algorithm may lead to inefficiency when dealing with huge data sets in terms of the dimensionality. Thus, a possible future work would be to investigate the parallelization approaches in the sense that the required memory to store the grid information could be distributed. As another option for the further investigation could be to enhance the partitioning procedure in the sense that it results a sparse grid which is cheaper in terms of the memory.

REFERENCES

[1] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. 1998. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD Conference*. 94–105.
[2] Paul E. Green Anil Chaturvedi and J. Douglas Caroll. 2001. K-modes Clustering. *Journal of Classification* 18, 1 (2001).
[3] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. OPTICS: Ordering Points to Identify the Clustering Structure *(SIGMOD '99)*. New York, NY, USA.
[4] Christian Böhm, Christos Faloutsos, Jia Pan, and Claudia Plant. 2006. Robust information-theoretic clustering. In *KDD*.
[5] A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39, 1 (1977), 1–38.
[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.. In *KDD Conference*.
[7] J. Macqueen. 1967. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*. 281–297.
[8] Joel D. Melo, Edgar M. Carreno, Aida Clavino, and Antonio Padilha-Feltrin. 2014. Determining spatial resolution in spatial load forecasting using a grid-based model. *Electric Power Systems Research* 111 (2014), 177–184.
[9] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. 2001. On Spectral Clustering: Analysis and an Algorithm *(NIPS'01)*.
[10] Jorma Rissanen. 2005. *An Introduction to the MDL Principle*. Technical Report. Helsinkin Institute for Information Technology.
[11] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. 1998. WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases *(VLDB '98)*. San Francisco, CA, USA.
[12] R. Sibson. 1973. SLINK: An optimally efficient algorithm for the single-link cluster method. *Comput. J.* 16, 1 (1973), 30–34.
[13] Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary?. In *ICML Conference'09*.
[14] Wei Wang, Jiong Yang, and Richard R. Muntz. 1997. STING: A Statistical Information Grid Approach to Spatial Data Mining *(VLDB '97)*. San Francisco, CA, USA, 186–195.