Advances in Intelligent Systems and Computing 245

Van-Nam Huynh · Thierry Denœux
Dang Hung Tran · Anh Cuong Le
Son Bao Pham *Editors*

Knowledge and Systems Engineering

Proceedings of the Fifth International Conference KSE 2013, Volume 2

Springer

Advances in Intelligent Systems and Computing

Volume 245

Series Editor

Janusz Kacprzyk, Warsaw, Poland

For further volumes:
http://www.springer.com/series/11156

Van-Nam Huynh · Thierry Denœux
Dang Hung Tran · Anh Cuong Le
Son Bao Pham
Editors

Knowledge and Systems Engineering

Proceedings of the Fifth International Conference KSE 2013, Volume 2

Springer

Editors
Van-Nam Huynh
School of Knowledge Science
Japan Advanced Institute of Science and Technology
Ishikawa
Japan

Thierry Denœux
Universite de Technologie de Compiegne
Compiegne Cedex
France

Dang Hung Tran
Faculty of Information Technology
Hanoi National University of Education
Hanoi
Vietnam

Anh Cuong Le
Faculty of Information Technology
University of Engineering and Technology - VNU Hanoi
Hanoi
Vietnam

Son Bao Pham
Faculty of Information Technology
University of Engineering and Technology - VNU Hanoi
Hanoi
Vietnam

ISSN 2194-5357 ISSN 2194-5365 (electronic)
ISBN 978-3-319-02820-0 ISBN 978-3-319-02821-7 (eBook)
DOI 10.1007/978-3-319-02821-7
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013950935

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains papers presented at the Fifth International Conference on Knowledge and Systems Engineering (KSE 2013), which was held in Hanoi, Vietnam, during 17–19 October, 2013. The conference was jointly organized by Hanoi National University of Education and the University of Engineering and Technology, Vietnam National University. The principal aim of KSE Conference is to bring together researchers, academics, practitioners and students in order to not only share research results and practical applications but also to foster collaboration in research and education in Knowledge and Systems Engineering.

This year we received a total of 124 submissions. Each of which was peer reviewed by at least two members of the Program Committee. Finally, 68 papers were chosen for presentation at KSE 2013 and publication in the proceedings. Besides the main track, the conference featured six special sessions focusing on specific topics of interest as well as included one workshop, two tutorials and three invited speeches. The kind cooperation of Yasuo Kudo, Tetsuya Murai, Yasunori Endo, Sadaaki Miyamoto, Akira Shimazu, Minh L. Nguyen, Tzung-Pei Hong, Bay Vo, Bac H. Le, Benjamin Quost, Sébastien Destercke, Marie-Hélène Abel, Claude Moulin, Marie-Christine Ho Ba Tho, Sabine Bensamoun, Tien-Tuan Dao, Lam Thu Bui and Tran Dinh Khang in organizing these special sessions and workshop is highly appreciated.

As a follow-up of the Conference, two special issues of the Journal of *Data & Knowledge Engineering* and *International Journal of Approximate Reasoning* will be organized to publish a small number of extended papers selected from the Conference as well as other relevant contributions received in response to subsequent calls. These journal submissions will go through a fresh round of reviews in accordance with the journals' guidelines.

We would like to express our appreciation to all the members of the Program Committee for their support and cooperation in this publication. We would also like to thank Janusz Kacprzyk (Series Editor) and Thomas Ditzinger (Senior Editor, Engineering/Applied Sciences) for their support and cooperation in this publication.

Last, but not the least, we wish to thank all the authors and participants for their contributions and fruitful discussions that made this conference a success.

Hanoi, Vietnam
October 2013

Van-Nam Huynh
Thierry Denœux
Dang Hung Tran
Anh Cuong Le
Son Bao Pham

Organization

Honorary Chairs

Van Minh Nguyen – Hanoi National University of Education, Vietnam
Ngoc Binh Nguyen – VNU University of Engineering and Technology, Vietnam

General Chairs

Cam Ha Ho – Hanoi National University of Education, Vietnam
Anh Cuong Le – VNU University of Engineering and Technology, Vietnam

Program Chairs

Van-Nam Huynh – Japan Advanced Institute of Science and Technology, Japan
Thierry Denœux – Université de Technologie de Compiègne, France
Dang Hung Tran – Hanoi National University of Education, Vietnam

Program Committee

Akira Shimazu, Japan
Azeddine Beghdadi, France
Son Bao Pham, Vietnam
Benjamin Quost, France
Bernadette Bouchon-Meunier, France
Binh Thanh Huynh, Vietnam
Bay Vo, Vietnam
Cao H, Tru, Vietnam
Churn-Jung Liau, Taiwan
Dinh Dien, Vietnam
Claude Moulin, France
Cuong Nguyen, Vietnam
Dritan Nace, France
Duc Tran, USA
Duc Dung Nguyen, Vietnam
Enrique Herrera-Viedma, Spain
Gabriele Kern-Isberner, Germany
Hiromitsu Hattori, Japan
Hoang Truong, Vietnam
Hung V. Dang, Vietnam
Hung Son Nguyen, Poland
Jean Daniel Zucker, France

Jérôme Lang, France
Jing Liu, China
Jiuyong Li, Australia
Jonathan Lawry, UK
Kenji Satou, Japan
Lam T. Bui, Vietnam
Bac H. Le, Vietnam
Loannis Parissis, France
Marie-Helene Abel, France
Martin Steffen, Norway
Masahiro Inuiguchi, Japan
Michel Riveill, France
Mina Ryoke, Japan
Minh-Dung Phan, Thailand
Mitsuru Ikeda, Japan
Minh L. Nguyen, Japan
Noboru Takagi, Japan
Peter Whigham, New Zealand
Phayung Meesad, Thailand
Quang-Huy Nguyen, France
Quang Uy Nguyen, Ireland
Sabine Bensamoun, France
Sadaaki Miyamoto, Japan
Serge Stinckwich, France
Sébastien Destercke, France
Si Quang Le, UK
Son Doan, USA
Tien-Tuan Dao, France
Tetsuya Murai, Japan
Thanh Binh Nguyen, Vietnam
Thanh Tri Nguyen, Vietnam
Thanh-Thuy Nguyen, Vietnam
The Duy Bui, Vietnam
The Loc Nguyen, Vietnam
Thomas Huynh, USA
Tho Hoan Pham, Vietnam
Thepchai Supnithi, Thailand
The Dung Luong, Vietnam
Tran Dinh Khang, Vietnam
Tsutomu Fujinami, Japan
Tzung-Pei Hong, Taiwan
Vladik Kreinovich, USA
Xiaoshan Li, Macau
Xuan Hoai Nguyen, Vietnam
Xuan-Hieu Phan, Vietnam
Yasuo Kudo, Japan

Contents

Part I: Workshop Invited Talks

Part II: KSE 2013 Special Sessions and Workshop

Erratum

Part I
Workshop Invited Talks

The Place of Causal Analysis in the Analysis of Simulation Data

Ladislav Hluch

Abstract. This talk briefly reviews selected basic concepts and principles of structural approach to causal analysis, and outlines how they could be harnessed for analyzing and summarizing the data from simulations of complex dynamic systems, and for exploratory analysis of simulation models through machine learning. We illustrate the proposed method in the context of human behaviour modeling on a sample scenario from the EDA project A-0938-RT-GC EUSAS. The method revolves around the twin concepts of a causal partition of a variable of interest, and a causal summary of a simulation run. We broadly define a causal summary as a partition of the significant values of the analyzed variables (in our case the simulated motives fear and anger of human beings) into separate contributions by various causing factors, such as social influence or external events. We demonstrate that such causal summaries can be processed by machine learning techniques (e.g. clustering and classification) and facilitate meaningful interpretations of the emergent behaviours of complex agent-based models.

Acknowledgement. This work was supported by the European Defence Agency project A-0938-RT-GC EUSAS, by the Slovak Research and Development Agency under the contract No. APVV-0233-10, and by the project VEGA No. 2/0054/12.

Ladislav Hluch
Institute of Informatics, Slovak Academy of Sciences

V.-N. Huynh et al. (eds.), *Knowledge and Systems Engineering, Volume 2*,
Advances in Intelligent Systems and Computing 245,
DOI: 10.1007/978-3-319-02821-7_1, © Springer International Publishing Switzerland 2014

Evolutionary Computation in the Real World: Successes and Challenges

Graham Kendall

Abstract. Evolutionary Computation has the potential to address many problems which may seem intractable to some of the methodologies that are available today. After briefly describing what evolutionary computation is (and what it is not), I will outline some of the success stories before moving onto the challenges we face in having these algorithms adopted by the industrial community at large.Some of the areas I will draw upon include Checkers and Chess, Scheduling and Timetabling, Hyper-heuristics and Meta-heuristics, as well some other problems drawn from the Operational Research literature.

Graham Kendall
The University of Nottingham Malaysia Campus,
Selangor Darul Ehsan, Malaysia

V.-N. Huynh et al. (eds.), *Knowledge and Systems Engineering, Volume 2*,
Advances in Intelligent Systems and Computing 245,
DOI: 10.1007/978-3-319-02821-7_2, © Springer International Publishing Switzerland 2014

Part II
KSE 2013 Special Sessions and Workshop

A Method of Two-Stage Clustering with Constraints Using Agglomerative Hierarchical Algorithm and One-Pass k-Means++

Yusuke Tamura, Nobuhiro Obara, and Sadaaki Miyamoto

Abstract. The aim of this paper is to propose a two-stage method of clustering in which the first stage uses one-pass k-means++ and the second stage uses an agglomerative hierarchical algorithm. This method outperforms a foregoing two-stage algorithm by replacing the ordinary one-pass k-means by one-pass k-means++ in the first stage. Pairwise constraints are also taken into consideration in order to improve its performance. Effectiveness of the proposed method is shown by numerical examples.

1 Introduction

Clustering techniques [7, 9] has recently been becoming more and more popular, as huge data on the web should be handled. Such data are frequently unclassified in contrast to those in traditional pattern classification problems where most data have classification labels [5]. Not only methods of unsupervised classification but also those of semi-supervised classification [6] and constrained clustering [2, 3] have been developed to handle such data.

Clustering techniques in general can be divided into two categories of hierarchical clustering and non-hierarchical clustering. Best-known methods in the first category are agglomerative hierarchical clustering, while that in the second category is the method of k-means [8]. Most methods of semi-supervised classification and constrained clustering are non-hierarchical, but agglomerative hierarchical clustering is at least as useful as non-hierarchical techniques in various applications. A drawback in agglomerative hierarchical clustering is that larger computation is needed when compared with simple non-hierarchical methods such as the k-means.

Yusuke Tamura · Nobuhiro Obara
Master's Program in Risk Engineering, University of Tsukuba, Ibaraki 305-8573, Japan

Sadaaki Miyamoto
Department of Risk Engineering, University of Tsukuba, Ibaraki 305-8573, Japan
e-mail: miyamoto@risk.tsukuba.ac.jp

V.-N. Huynh et al. (eds.), *Knowledge and Systems Engineering, Volume 2*,
Advances in Intelligent Systems and Computing 245,
DOI: 10.1007/978-3-319-02821-7_3, © Springer International Publishing Switzerland 2014

Here is a question: how can we develop a method of agglomerative hierarchical clustering that can handle large amount of data with semi-supervision or constraints? We have partly answered this question by developing a method of agglomerative hierarchical clustering in which pairwise constraints can be handled using penalties in the agglomerative clustering algorithm [11]. Moreover a two-stage clustering has been suggested in which the first-stage uses k-means and the second stage is a class of agglomerative hierarchical clustering [10]. However, performance of the two-stage algorithm should still be improved.

In this paper we introduce a variation of the algorithm presented in [10]. In short, we use one-pass k-means++[1] in the first stage and show an improved two stage clustering algorithm with pairwise constraints. Several numerical examples are shown to observe the usefulness of the proposed method.

The rest of this paper is organized as follows. Section 2 provides preliminaries, then Section 3 shows the two-stage algorithm herein. Section 4 shows effectiveness and efficiency of the proposed algorithm using a number of numerical examples. Finally, Section 5 concludes the paper.

2 Preliminary Consideration

We begin with notations. Let the set of objects be $X = \{x_1, \cdots, x_n\}$. Each object x_k is a point in the p-dimensional Euclidean space $\boldsymbol{R}^p$: $x_i = (x_{i1}, \cdots, x_{ip}) \in \boldsymbol{R}^p$

Clusters are denoted by $G_1, G_2, \cdots, G_C$, and the collection of clusters is given by $\mathscr{G} = \{G_1, G_2, \cdots, G_C\}$. Clusters are partition of X:

$$\bigcup_{i=1}^{C} G_i = X,\ G_i \cap G_j = \emptyset\ (i \neq j) \tag{1}$$

2.1 Agglomerative Hierarchical Clustering

Assume that $d(G, G')$ is a dissimilarity measure defined between two clusters; calculation formula of $d(G, G')$ will be given after the following general algorithm of agglomerative hierarchical clustering, abbreviated **AHC** in which **AHC 1** and **AHC 2** are the steps of this algorithm.

AHC1: Let initial clusters given by objects.
$G_i = \{x_i\}, (i = 1, \cdots, n)$
$C = n$, (C is the number of clusters and n is the number of objects)
Calculate $d(G, G')$ for all pairs $G, G' \in \mathscr{G} = \{G_1, G_2, \cdots, G_C\}$.
AHC2: Merge the pair of clusters of minimum dissimilarity:

$$d(G_q, G_r) = \arg \min_{G, G' \in \mathscr{G}} d(G, G') \tag{2}$$

```
    Add $\hat{G} = G_q \cup G_r$ to $\mathscr{G}$ and remove $G_q, G_r$ from $\mathscr{G}$.
    $C = C - 1$ .
    If $C = 1$, then output the process of merge of clusters as a dendrogram and stop.
AHC3: Calculate $d(\hat{G}, G')$ for $\hat{G}$ and all other $G' \in \mathscr{G}$. go to **AHC2**.
```

We assume that the dissimilarity between two objects is given by the squared Euclidean distance:

$$d(x_k,x_l) = \|x_k - x_l\|^2 = \sum_{j=1}^{p} (x_{kj} - x_{lj})^2.$$

Moreover the centroid method is used here, which calculate $d(\hat{G}, G')$ as follows.

Centroid method:

Let $M(G)$ be the centroid (the center of gravity) of G:

$$M(G) = (M_1(G), \cdots, M_p(G))^T,$$

where

$$M_j(G) = \frac{1}{|G|} \sum_{x_k \in G} x_{kj}, \ (j = 1, \cdots, p) \tag{3}$$

and let

$$d(G, G') = \|M(G) - M(G')\|^2 \tag{4}$$

2.2 *k*-Means and *k*-Means++

The method of k-means repeats the calculation of centroids of clusters and nearest centroid allocation of each object until convergence [4]. It has been known that the result is strongly dependent on the choice of initial values.

The method of k-means++ [1] improves such dependence on initial clusters by using probabilistic selection of initial centers. To describe k-means++, let v_i be the i-th cluster center and $D(x)$ be the Euclidean distance between object x and the already selected centers nearest to x. The algorithm is as follows [1].

```
1a: Let the first cluster center $v_1$ be a randomly selected object from $X$.
1b: Let a new center $v_i$ be selected from $X$ with probability $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$.
1c: Repeat **1b** until $k$ cluster centers are selected.
2: Carry out the ordinary $k$-means algorithm.
```

Step **1b** is called “D^2 weighting”, whereby a new cluster center that have larger distance from already selected centers will have larger probability to be selected.

2.3 Pairwise Constraints

Two sets ML and CL of constraints are used in constrained clustering [2, 3]. A set $ML = \{(x_i, x_j)\} \subset X \times X$ consists of *must-link* pairs so that x_i and x_j should be in a same cluster, while another set $CL = \{(x_k, x_l)\} \subset X \times X$ consists of *cannot-link* pairs so that x_i and x_j should be in different clusters. ML and SL are assumed to be symmetric in the sense that if $(x_i, x_j) \in ML$ then $(x_j, x_i) \in ML$, and if $(x_k, x_l) \in CL$ then $(x_l, x_k) \in CL$.

Note that ML is regarded as an undirected graph in which nodes are objects appeared in ML, and an undirected edge is $(x_i, x_j) \in ML$.

Introduction of the pairwise constraints to k-means has been done by Wagstaff et al. [12]. The developed algorithm is called COP k-means.

3 A Two-Stage Algorithm

A two-stage algorithm of clustering for large-scale data is proposed, in which the first stage uses one-pass k-means++ to have a medium number of cluster centers and the second stage uses the centroid method. Pairwise constraints are taken into account in both stages.

3.1 One-Pass COP *k*-Means++

One pass k-means implies that the algorithm does not iterate the calculation of the centroid and the nearest center allocation: it first generates initial cluster centers, then each object is allocated to the cluster of the nearest center. After the allocation, new cluster centers are calculated as the centroids (3). Then the algorithm stops without further iteration.

Pairwise Constraints in the First Stage

Moreover the one-pass algorithm must take pairwise constraints into account. ML (must-link) is handled as the initial set of objects, as ML defines a connected components of a graph. Then the centroid of the connected components is used instead of the objects in the components. On the other hand, CL (cannot-link) is handled in the algorithm.

Thus the algorithm in the first stage is called one-pass COP k-means++, which is as follows.

One-Pass COP k-means++ in the first stage

1: Let initial clusters be generated by using the D^2 weighting.

2: Each object $x \in X$ is allocated to the cluster of the nearest center that does not break the given pairwise constraints CL. If x cannot be allocated to any cluster due to the constraints, stop with flag **FAILURE**.

3: Cluster centers are updated as the centroids (3).

4: Stop. (Note that this step is replaced by 'repeat steps 2 and 3 until convergence' if the one-pass condition is removed.)
End of One-Pass COP k-means++.

3.2 Agglomerative Algorithm in the Second Stage

Information of the centroids $M(G_i)$ and the number of elements $|G_i|$ in cluster G_i $(i = 1, 2, \ldots, c)$ is passed to the second stage. Note that information concerning every object $x \in X$ is not required to generate clusters by AHC.

Different sets of $M(G_i)$ are obtained from the first stage. To have better clusters in the second stage, a number of different trials of the first stage are made and those centroids with the minimum value of

$$J = \sum_{i=1}^{C} \sum_{x \in G_i} \|x - M(G_i)\|^2 \tag{5}$$

is taken for the second stage.

Pairwise Constraints in the Second Stage

Although must-link constraints is already handled in the first stage, cannot-link constraints still exist in the second stage. Hence CL is handled by a penalty term in the following algorithm.

Penalized Agglomerative Hierarchical Clustering Algorithm (P-AHC)

```
P-AHC1: For initial clusters derived from the first stage, calculate $d(G,G')$ for
    all $G,G' \in \mathscr{G}$.
P-AHC2:
    $d(G_q,G_r) = \arg\min_{G,G' \in \mathscr{G}} \{d(G,G') + \sum_{x_k \in G, x_l \in G'} \omega_{kl}\}$
    using the penalty term with $\omega_{kl}$:
    if $(x_k,x_l) \in CL$, $\omega_{kl} > 0$; if $(x_k,x_l) \notin CL$, $\omega_{kl} = 0$.
    Let $\bar{G} = G_q \cup G_r$.
    Add $\bar{G}$ to $\mathscr{G}$ and delete $G_q, G_r$ from $\mathscr{G}$.
    $C = C - 1$. If $C = 1$, stop.
P-AHC3: Calculate $d(\bar{G},G')$ for all other $G' \in \mathscr{G}$. Go to P-AHC2.
```

Note that ω is taken to be sufficient large, i.e., we assume hard constraints.

4 Numerical Examples

Two data sets were used for evaluating the present method with other methods already proposed elsewhere. One is an artificial data set on the plane, while the second is a real data set from a data repository [1].

As for the methods, the following abbreviated symbols are used:

- PAHC: penalized AHC algorithm;
- COPKPP: one-pass COP *k*-means++ ;
- COPK: ordinary one-pass COP *k*-means ;
- COPKPP(n): one-pass COP *k*-means++ with n different initial values;
- COPK(n): one-pass COP *k*-means with n different initial values.

The computation environment is as follows.

CPU: Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz - 3.60GHz
Memory: 8.00 GB
OS: Windows 7 Professional 64bit
Programming Language: C

Two Circles

First data is shown In Fig. 1. The objective is to separate the outer circle having 700 points and the inner circle with 9,300 points. Note that the two clusters are 'unbalanced' in the sense that the numbers of objects are very different.

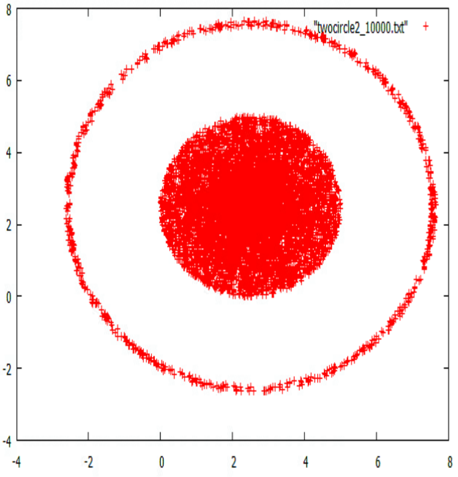

Fig. 1 Data of 'two circles'

Shuttle Data Set

The Shuttle data set downloaded from [1] has 9 dimensions that can be divided into seven classes. About 80% of points belong to Class 1. We divide this data set into two clusters: one cluster is Class 1 and another cluster should be other six classes, since to detect small six clusters in 20% of points and one large cluster of 80% of points directly is generally a difficult task.

Evaluation Criteria

The evaluation has been done using three criteria: objective function values, the Rand index, and the run time.

Note that CL alone is used and ML is not used here, since ML was found to be not useful when compared with CL by preliminary tests on these data sets.

Pairs of objects in CL were randomly selected from the data set: one object from a cluster and another object from another cluster. For artificial data set the number in CL varies from 0 to 50; for the Shuttle data the number in CL varies from 0 to 500. The number of trials $n = 100$ (the number of trials in the first stage is 100) or $n = 10$ were used.

4.1 Evaluation by Objective Function Value

The averages of objective function values J are plotted in Figs. 2 and 3, respectively for the artificial data and the Shuttle data.

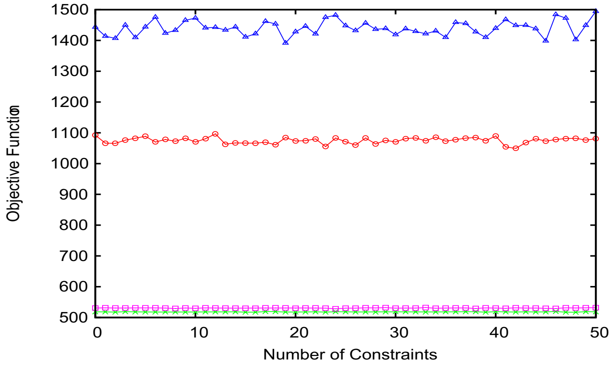

Fig. 2 Objective function values with CL for artificial data. Red circles are for COPK(100)-PAHC. Green × are for COPKPP(100)-PAHC. Blue triangles are for COPK(10)-PAHC. Pink squares are for COPKPP(10)-PAHC.

From these figures it is clear that COPKPP-PAHC has less values of the objective function than COPK-PAHC.

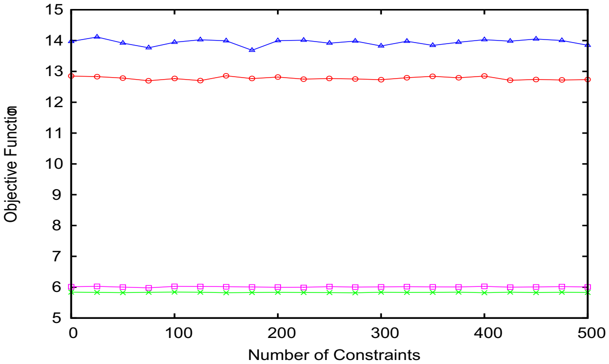

Fig. 3 Objective function values with *CL* for the Shuttle data. Red circles are for COPK(100)-PAHC. Green × are for COPKPP(100)-PAHC. Blue triangles are for COPK(10)-PAHC. Pink squares are for COPKPP(10)-PAHC.

4.2 Evaluation by RandIndex

The Rand index has been used as a standard index to measure precision of classification [12]:

$$Rand(P_1,P_2) = \frac{|C_a|+|C_b|}{{}_nC_2} \tag{6}$$

where P_1 and P_2 means the precise classification and the actually obtained classification. $|C_a|$ is the number of pairs of objects in C_a such that a pair in C_a is in the same precise class and at the same time in the same cluster obtained by the experiment; $|C_b|$ is the number of pairs of objects in C_b such that a pair in C_a is in different precise classes and at the same time in different clusters obtained by the

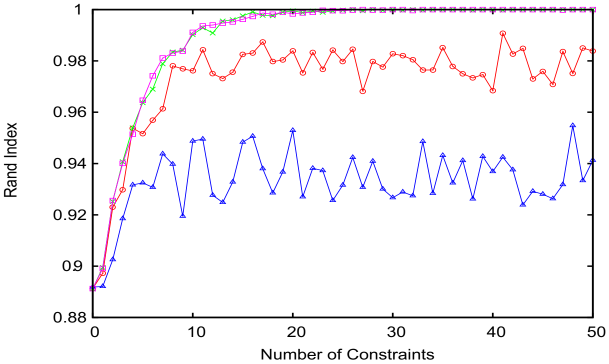

Fig. 4 Rand index values with *CL* for artificial data. Red circles are for COPK(100)-PAHC. Green × are for COPKPP(100)-PAHC. Blue triangles are for COPK(10)-PAHC. Pink squares are for COPKPP(10)-PAHC.

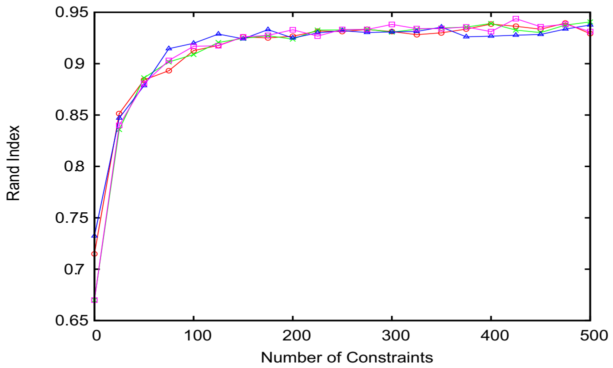

Fig. 5 Rand index values with *CL* for the Shuttle data. Red circles are for COPK(100)-PAHC. Green × are for COPKPP(100)-PAHC. Blue triangles are for COPK(10)-PAHC. Pink squares are for COPKPP(10)-PAHC.

experiment. If the resulting clusters precisely coincide with the precise classes, then $Rand(P_1,P_2) = 1$, and vice versa.

The Rand index with $n = 100$ has been calculated and the results are shown in Figs. 4 and 5, respectively for the artificial data and the Shuttle data. The former figure shows advantage of COPKPP, while the effect of K-means++ is not clear in the second example.

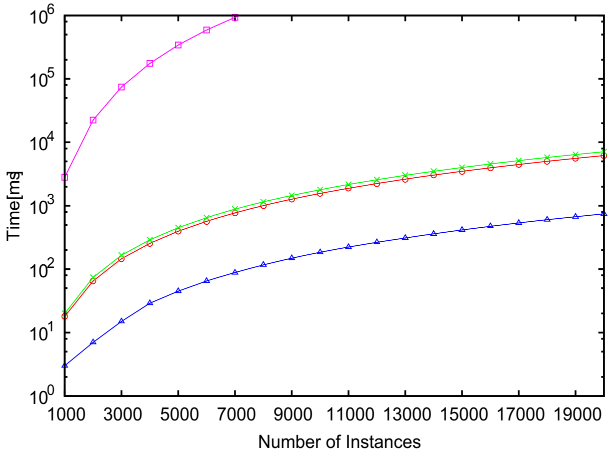

Fig. 6 Relation between the number of objects in artificial data and the CPU time. Red circles are for COPK(100)-PAHC. Green × are for COPKPP(100)-PAHC. Blue triangles are for COPKPP(10)-PAHC. Pink squares are for PAHC.

4.3 Evaluation by CPU Time

How total CPU time varies by using one-pass COP k-means++ or one-pass COP k-means was investigated. The used methods were COPK(100)-PAHCCCOPKPP(100)-PAHCCCOPKPP(10)-PAHCC and PAHC (without the first stage). Ten trials with n objects and their average CPU time was measured with $n = 1,000 - 20,000$. In the first stage the number of objects was reduced to 1% and the second stage AHC was carried out. The result is shown in Fig. 6.

Fig. 6 shows that CPU time was reduced to 0.1% by introducing the two-stage method. When COPK(100)-PAHC and COPKPP(100)-PAHC are comparted, the latter needs more time, but the difference is not notable.

5 Conclusion

This paper proposed a two-stage algorithm in which the first stage uses one-pass k-means++ and the second stage uses the centroid method of agglomerative hierarchical clustering. Pairwise constraints were moreover introduced in the algorithm. It has been shown by numerical examples that one-pass k-means++ is effective when compared with one-pass k-means in the first stage. Thus the dependence on initial values was greatly improved. Moreover the use of cannot-links was effective in the numerical examples. This inclination is in accordance with other studies, e.g., [11].

The two-stage procedure could handle relatively large-scale data sets. However, more tests on larger real data should be done as a future work in order to show the usefulness of the proposed method in a variety of applications.

Acknowledgment. The authors greatly appreciate anonymous reviewers for their useful comments. This study has partially been supported by the Grant-in-Aid for Scientific Research, JSPS, Japan, No.23500269.

References

1. Arthur, D., Vassilvitskii, S.: k-means++: The Advantages of Careful Seeding. In: Proc. of SODA 2007, pp. 1027–1035 (2007)
2. Basu, S., Bilenko, M., Mooney, R.J.: A Probabilistic Framework for Semi-Supervised Clustering. In: Proc. of the Tenth ACM SIGKDD (KDD 2004), pp. 59–68 (2004)
3. Basu, S., Davidson, I., Wagstaff, K.L. (eds.): Constrained Clustering. CRC Press (2009)
4. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)
5. Bishop, C.: Pattern Recognition and Machine Learning. Springer (2006)
6. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press (2006)
7. Everitt, B.S.: Cluster Analysis, 3rd edn., Arnold (1993)
8. MacQueen, J.B.: Some methods of classification and analysis of multivariate observations. In: Proc. of 5th Berkeley Symposium on Math. Stat. and Prob., pp. 281–297 (1967)
9. Miyamoto, S.: Introduction to Cluster Analysis. Morikita-shuppan (1999) (in Japanese)

10. Obara, N., Miyamoto, C.S.: A Method of Two-Stage Clustering with Constraints Using Agglomerative Hierarchical Algorithm and One-Pass *K*-Means. In: Proc. of SCIS-ISIS 2012, pp. 1540–1544 (2012)
11. Terami, A., Miyamoto, S.: Constrained Agglomerative Hierarchical Clustering Algorithms with Penalties. In: Proc. of FUZZ-IEEE 2011, pp. 422–427 (2011)
12. Wagstaff, N., Cardie, C., Rogers, S., Schroedl, S.: Constrained K-means Clustering with Background Knowledge. In: Proc. of ICML 2001, pp. 577–584 (2001)
13. `http://archive.ics.uci.edu/ml/`

An Algorithm Combining Spectral Clustering and DBSCAN for Core Points

So Miyahara, Yoshiyuki Komazaki, and Sadaaki Miyamoto

Abstract. The method of spectral clustering is based on the graph Laplacian, and outputs good results for well-separated groups of points even when they have nonlinear boundaries. However, it is generally difficult to classify a large amount of data by this technique because computational complexity is large. We propose an algorithm using the concept of core points in DBSCAN. This algorithm first applies DBSCAN for core points and performs spectral clustering for each cluster obtained from the first step. Simulation examples are used to show performance of the proposed algorithm.

1 Introduction

Many researchers are now working on analysis of huge data on the web. In accordance with this, many methods of data analysis have been developed. Data clustering is not exceptional: nowadays a variety of new algorithms of clustering is being applied to large-scale data sets. Special attention has been paid to spectral clustering [4, 2, 3] which is based on a weighted graph model and uses the graph Laplacian. It has been known that this method works well even when clusters have strongly nonlinear boundaries between clusters, as far as they are well-separated.

In spite of its usefulness, the spectral clustering has a drawback: it has a relatively large computation when compared with a simple algorithm of the K-means [4, 5]. The latter can be applied to huge data, since the algorithm is very simple, but the former uses eigenvalues and eigenvectors which needs much more computation.

This paper proposes a method combining the spectral clustering and the idea in a simple graph-theoretical method based on DBSCAN [6]. The both methods are

So Miyahara · Yoshiyuki Komazaki
Master's Program in Risk Engineering, University of Tsukuba, Ibaraki 305-8573, Japan

Sadaaki Miyamoto
Department of Risk Engineering, University of Tsukuba, Ibaraki 305-8573, Japan
e-mail: miyamoto@risk.tsukuba.ac.jp

V.-N. Huynh et al. (eds.), *Knowledge and Systems Engineering, Volume 2*,
Advances in Intelligent Systems and Computing 245,
DOI: 10.1007/978-3-319-02821-7_4, © Springer International Publishing Switzerland 2014

well-known, but their combination with a simple modification leads a new algorithm. A related study has been done by Yan et al. [7] in which K-means is first used and the centers from K-means are clustered using the spectral clustering. The present study is different from [7], since the original objects are made into clusters by the spectral clustering by the method herein, whereas the K-means centers are clustered in [7]. A key point is that only core-points are used for clustering, and other 'noise points' are allocated to clusters using a simple technique of supervised classification. Moreover, these two methods of the spectral clustering and DBSCAN has a common theoretical feature that is useful for reducing computation, and hence the combination proposed here has a theoretical basis, as we will see later. Such a feature cannot be found between K-means and the spectral clustering.

The rest of this paper is organized as follows. Section 2 gives preliminaries, and then Section 3 proposes a new algorithm using the spectral clustering and DBSCAN for core points. Section 4 shows illustrative examples and a real example. Finally, Section 5 concludes the paper.

2 Preliminary Consideration

This section discusses the well-known methods of the spectral clustering and DBSCAN.

2.1 Spectral Clustering

The spectral clustering, written as SC here, uses a partition of a graph of objects $D = \{1, 2, \ldots, n\}$ for clustering. The optimality of the partition is discussed in [3] but omitted here.

Assume that the number of clusters is fixed and given by c. A similarity matrix $S = (s_{ij})$ is generated using a dissimilarity $d(i, j)$ between i and j. We assume that $d(i, j)$ is the Euclidean distance in this paper, although many other dissimilarity can also be used for the same purpose.

$$S = [s_{ij}],\ s_{ij} = \exp\left(-\frac{d(i,j)}{(2\sigma^2)}\right)$$

where σ is a positive constant. When the ε-neighborhood graph should be used, then those s_{ij} with $d(i, j) > \varepsilon$ should be set to zero. We then calculate

$$D = diag(d_1, \cdots, d_n),\ d_i = \sum_{j=1}^{n} s_{ij}$$

and the graph Laplacian L:

$$L = D^{-\frac{1}{2}}(D - S)D^{-\frac{1}{2}}$$

Minimum c eigenvalues are taken and the corresponding eigenvectors are assumed to be $u_1, \cdots, u_c$. A matrix

$$U = (u_1, \cdots, u_c)$$

is then defined. Each component of the eigenvalues has correspondence to an object. Then K-means clustering of each rows with c clusters will give the results of clustering by SC [3]. Concretely, suppose row vectors of U are $\boldsymbol{u}_1^\top, \ldots, \boldsymbol{u}_n^\top$: $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n)^\top$, then K-means algorithm is applied to objects $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$, where $\boldsymbol{u}_j$ $(j = 1, \ldots, n)$ is a c-vector [3].

2.2 DBSCAN-CORE

DBSCAN proposed by Ester et al. [6] generates clustering based on density of objects using two parameters Eps and MinPts. For given Eps and MinPts, the Eps-neighborhood of $p \in D$ is given by

$$N_{\mathrm{Eps}}(p) = \{q \in D \mid d(p,q) \leq \mathrm{Eps}\}$$

When an object p satisfies $|N_{\mathrm{Eps}}(p)| \geq$ MinPts, then p is called a core-point (note: $|N_{\mathrm{Eps}}(p)|$ is the number of elements in $N_{\mathrm{Eps}}(p)$).

If the next two conditions are satisfied, then p is called *directly density-reachable from q*:

1. $p \in N_{\mathrm{Eps}}(q)$, and
2. $|N_{\mathrm{Eps}}(q)| \geq$ MinPts (q is a core-point).

A variation of the DBSCAN algorithm used here starts from a core-point called seed, and then collects all *core points* that are directly density-reachable from the seed. Then they form a cluster. Then the algorithm repeats the same procedure until no more cluster is obtained. The remaining objects are left unclassified. In other words, this algorithm searches the connected components of the graph generated from core points with the edges of direct reachability, and defines clusters as the connected components.

This algorithm is simpler than the original DBSCAN in that only core-points are made into clusters, while non-core points are included in clusters by the original DBSCAN. Therefore the present algorithm is called DBSCAN-CORE in this paper. Specifically, The set D is first divided into C of core points and N of non-core points:

$$D = C \cup N, \qquad C \cap N = \emptyset.$$

Clusters $C_1, \ldots, C_l$ generated by DBSCAN-CORE is a partition of C:

$$\bigcup_{i=1}^{l} C_i = C, \qquad C_i \cap C_j = \emptyset \ \ (i \neq j).$$

How to decide appropriate values of the parameters is given in [6], but omitted here.