Learning Graph Representation via Formal Concept Analysis

Yuka Yoneda
ISIR, Osaka University
yoneda@ar.sanken.osaka-u.ac.jp

Mahito Sugiyama
National Institute of Informatics
JST, PRESTO
mahito@nii.ac.jp

Takashi Washio
ISIR, Osaka University
washio@ar.sanken.osaka-u.ac.jp

Abstract

We present a novel method that can learn a *graph representation* from multivariate data. In our representation, each node represents a cluster of data points and each edge represents the subset-superset relationship between clusters, which can be mutually overlapped. The key to our method is to use *formal concept analysis* (FCA), which can extract hierarchical relationships between clusters based on the algebraic closedness property. We empirically show that our method can effectively extract hierarchical structures of clusters compared to the baseline method.

1 Introduction

Representation learning has become one of the most important tasks in machine learning (Bengio et al., 2013; Goodfellow et al., 2016). The typical task is to find a numerical (vectorized) representation from structured objects, such as images (Krizhevsky et al., 2012), speeches (Graves et al., 2013), and texts (Mikolov et al., 2013), which have discrete structures between variables. However, to date, learning of a structured representation from numerical data has not been studied at sufficient depth, while the task is crucial for *relational reasoning* aiming at finding relationships between objects to bridge between a symbolic approach and a gradient-based numerical approach (Santoro et al., 2017).

A classic yet promising branch of research for learning structures from numerical data is *hierarchical clustering* (Maimon and Rokach, 2005), which is a widely used unsupervised learning method in multivariate data analysis from natural language processing (Brown et al., 1992) to human motion analysis (Zhou et al., 2013). Given a set of data points without any class labels, hierarchical clustering can learn a tree structured representation, called a *dendrogram*, whose nodes correspond to clusters and leaves correspond to the input data points. The resulting tree structures can be used for further analysis of relational reasoning and other machine learning tasks.

However, the representation in hierarchical clustering is restricted to the form of a *binary tree* since clusters must be disjoint with each other in the exiting approaches. Nevertheless, clusters of objects can often overlap in real-world data analysis. Therefore the technique that can find a hierarchical structure of overlapped clusters from multivariate data is needed, which leads to a more flexible *graph structured representation* of numerical data.

To solve the problem, we propose to combine *nearest neighbor-based binarization* and *formal concept analysis* (FCA) (Davey and Priestley, 2002). FCA can extract a hierarchical structure of data using the algebraic closedness property, which consists of mutually overlapped clusters (Valtchev et al., 2004). Since FCA is designed for binary data, we first binarize numerical data by nearest neighbor-based binarization, which can model local geometric relationships between data points.

The remainder of this paper is organized as follows. Section 2 introduces our method; Section 2.1 explains nearest neighbor based binarization and Section 2.2 introduces FCA. Section 3 empirically examines our method and Section 4 summarizes our contribution.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

2 The Proposed Method

We introduce our method that learns a graph representation from numerical data. It consists of two stages. It first binarizes numerical data by nearest neighbor-based binarization, followed by applying formal concept analysis (FCA) (Davey and Priestley, 2002) to the binarized data, which is an established method to analyze relational databases (Kaytoue et al., 2011). Input to our method is an unlabeled real-valued vectors. Let $D = \{\boldsymbol{d}_1, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_n\}$ be an input dataset. Each data point is an m-dimensional vector and is denoted by $\boldsymbol{d}_i = (d_i^1, d_i^2, \ldots, d_i^m) \in \mathbb{R}^m$.

2.1 Nearest Neighbor-based Binarization

In the first stage, we convert each m-dimensional vector $\boldsymbol{d}_i \in \mathbb{R}^m$ into an n-dimensional binary vector $\boldsymbol{z}_i \in \{0,1\}^n$, where n coincides with the number of data points. The jth feature in the converted binary vector $\boldsymbol{z}_i$ shows whether or not the jth data point $\boldsymbol{d}_j$ belongs to nearest neighbors of $\boldsymbol{d}_i$. Formally, given a dataset $D \subset \mathbb{R}^m$ and a parameter $k \in \mathbb{N}$, each data point $\boldsymbol{d}_i \in D$ is binarized to the binary vector $\boldsymbol{z}_i = (z_i^1, z_i^2, \ldots, z_i^n) \in \{0,1\}^n$, where each component z_i^j is defined as

$$z_i^j = \begin{cases} 1 & \text{if } \boldsymbol{d}_j \text{ is the } l\text{th nearest data point from } \boldsymbol{d}_i \text{ for } l \leq k, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Hence our binarization models local relationships between data points in terms of the relative closeness in the original feature space, which is often used as k-nearest neighbor graphs in spectral clustering (von Luxburg, 2007).

When we regard indices of data points as *items*, every binary vector $\boldsymbol{z}_i \in \{0,1\}^n$ can be directly treated as a *transaction* $X_i \subseteq \{1, 2, \ldots, n\}$ defined as $X_i = \{\, j \in \{1, 2, \ldots n\} \mid z_i^j = 1 \,\}$. In other words, X_i is the set of indices of the data points which are the lth nearest data points ($l \leq k$) from $\boldsymbol{d}_i$, and hence $|X_i| = k$ always holds. Output in this stage is the transaction database $\mathcal{T} = \{X_1, X_2, \ldots, X_n\}$. Transaction databases are the standard data format in frequent pattern mining (Aggarwal and Han, 2014) and other fields in databases.

2.2 Formal Concept Analysis

In the second stage, we apply formal concept analysis (FCA) (Davey and Priestley, 2002) to the transaction database obtained from the first stage, which is a mathematical way to analyze databases based on the lattice theory and can be viewed as a co-clustering method for binary data. FCA can obtain hierarchical relationships of the original numerical data via the binarized transaction database.

Let $\mathcal{A} \subseteq \mathcal{T}$ be a subset of transactions and $B \subseteq [n] = \{1, 2, \ldots, n\}$ be a subset of data indices. We define that $\mathcal{A}'$ is the set of indices common to all transactions in $\mathcal{A}$ and B' is the set of transactions possessing all indices in B; that is,

$$\mathcal{A}' = \{\, j \in [n] \mid j \in X_i \text{ for all } X_i \in \mathcal{A} \,\}, \quad B' = \{\, X_i \in \mathcal{T} \mid B \subseteq X_i \,\}.$$

The pair $(\mathcal{A}, B)$ is called a *concept* if and only if $\mathcal{A}' = B$ and $B' = \mathcal{A}$. Here the mapping $''$ is a *closure operator*, as it satisfies $\mathcal{A} \subseteq \mathcal{A}''$, $\mathcal{A} \subseteq \mathcal{C} \Rightarrow \mathcal{A}'' \subseteq \mathcal{C}''$, and $(\mathcal{A}'')'' = \mathcal{A}''$, and $\mathcal{A}$ is *closed* if and only if $(\mathcal{A}, B)$ is a concept. A concept $(\mathcal{A}_1, B_1)$ is less general than a concept $(\mathcal{A}_2, B_2)$ if $\mathcal{A}_1$ is contained in $\mathcal{A}_2$; that is, $(\mathcal{A}_1, B_1) \leq (\mathcal{A}_1, B_1) \iff \mathcal{A}_1 \subseteq \mathcal{A}_2$, where the relation "$\leq$" becomes a partial order. The *concept lattice* is the set of concepts equipped with the order $\leq$. Intuitively, concepts are representative clusters in the dataset.

From the set $\mathfrak{L}$ of concepts, we finally construct a graph representation $G = (V, E)$, where $V = \{S \subseteq D \mid (T(S), T(S)') \in \mathfrak{L}\}$ with $T(S) = \{X_i \in \mathcal{T} \mid \boldsymbol{d}_i \in S\}$ and a directed edge $(v, w) \in E$ exists if v covers w; that is, $v \subset w$ and $v \subseteq u \subset w \Rightarrow u = v$. Hence $v \subseteq w$ if and only if w is reachable from v. This graph coincides with the Hasse diagram of the concept lattice using the partial order $\leq$. We illustrate an example of a graph representation in Figure 1(**b**) obtained by our method from a dataset in Figure 1(**a**).

Since the set of concepts is equivalent to that of *closed itemsets* used in closed itemset mining (Pasquier et al., 1999), we can efficiently enumerate all concepts using a closed itemset mining algorithm such as LCM (Uno et al., 2004). Moreover, we can directly obtain more compact representations by pruning nodes with small clusters using *frequent closet itemset mining* as the frequency (or the support) of an itemset coincides with the size of a cluster.

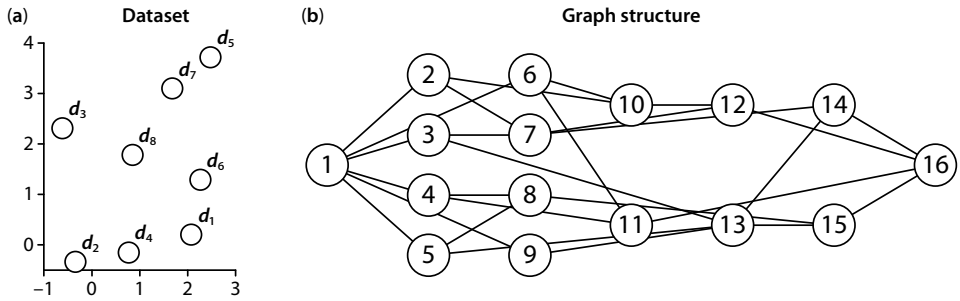

Figure 1: (**a**) Example of dataset. (**b**) Graph representation obtained from the dataset with $k = 3$. All edges are directed from left to right. Numbers of nodes indicate clusters as follows: 1: $\emptyset$, 2: $\{\boldsymbol{d}_1\}$, 3: $\{\boldsymbol{d}_6\}$, 4: $\{\boldsymbol{d}_3\}$, 5: $\{\boldsymbol{d}_8\}$, 6: $\{\boldsymbol{d}_2, \boldsymbol{d}_4\}$, 7: $\{\boldsymbol{d}_1, \boldsymbol{d}_6\}$, 8: $\{\boldsymbol{d}_3, \boldsymbol{d}_8\}$, 9: $\{\boldsymbol{d}_5, \boldsymbol{d}_7\}$, 10: $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_4\}$, 11: $\{\boldsymbol{d}_2, \boldsymbol{d}_3, \boldsymbol{d}_4\}$, 12: $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_4, \boldsymbol{d}_6\}$, 13: $\{\boldsymbol{d}_5, \boldsymbol{d}_6, \boldsymbol{d}_7, \boldsymbol{d}_8\}$, 14: $\{\boldsymbol{d}_1, \boldsymbol{d}_5, \boldsymbol{d}_6, \boldsymbol{d}_7, \boldsymbol{d}_8\}$, 15: $\{\boldsymbol{d}_3, \boldsymbol{d}_5, \boldsymbol{d}_6, \boldsymbol{d}_7, \boldsymbol{d}_8\}$, 16: $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3, \boldsymbol{d}_4, \boldsymbol{d}_5, \boldsymbol{d}_6, \boldsymbol{d}_7, \boldsymbol{d}_8\}$.

3 Experiments

We evaluate the proposed method using synthetic and real-world datasets. Since our method can be viewed as hierarchical clustering, to assess the effectiveness of our method, we compare our method with the standard hierarchical agglomerative clustering (HAC) with Ward's method (Ward Jr, 1963).

We performed all experiments on Windows10 Pro 64bit OS with a single processor of Intel Core i7-4790 CPU 3.60 GHz and 16GB of main memory. All experiments were conducted in Python 3.5.2. In our method, we used LCM (Uno et al., 2004) version 5.3[1] for closed itemset mining. HAC is implemented in scipy (Jones et al., 2001–).

We use dendrogram purity (DP) (Heller and Ghahramani, 2005) for evaluation. Dendrogram purity is the standard measure to evaluate the quality of hierarchical clusters (Kobren et al., 2017). Given a dataset $D = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_n\}$ and its ground truth partition $\mathcal{C} = \{C_1, C_2, \ldots, C_P\}$ such that $\bigcup_{C_i \in \{1,\ldots,P\}} C_i = D$ and $C_i \cap C_j = \emptyset$, and let $\mathcal{H}$ be a set of clusters obtained by a hierarchical clustering algorithm. We denote by $\mathrm{LCA}(\boldsymbol{d}_i, \boldsymbol{d}_j) \in \mathcal{H}$ the smallest cluster that includes both $\boldsymbol{d}_i$, $\boldsymbol{d}_j$ in $\mathcal{H}$ and $\mathrm{pur}(F, G) = |F \cap G|/|F|$ for a pair of clusters $F, G \subseteq D$. Assume that Q be the pair of data points in the same cluster; that is, $Q = \{(\boldsymbol{d}_i, \boldsymbol{d}_j) \mid \boldsymbol{d}_i, \boldsymbol{d}_j \in C_l \text{ for some } C_l \in \mathcal{C}\}$. The *dendrogram purity* of hierarchical clusters $\mathcal{H}$ is defined as

$$DP(\mathcal{H}) = \frac{1}{|Q|} \sum\nolimits_{l=1}^{P} \sum\nolimits_{\boldsymbol{d}_i, \boldsymbol{d}_j \in C_l} \mathrm{pur}(\mathrm{LCA}(\boldsymbol{d}_i, \boldsymbol{d}_j), C_l).$$

The dendrogram purity takes values from 0 to 1 and larger is better.

We use three types of synthetic datasets `synth1`, `synth2`, and `synth3`. `synth1` consists of two equal sized clusters sampled from two normal distributions $(\mu_0, \sigma_0^2) = (0, 1)$ and $(\mu_1, \sigma_1^2) = (2, 1)$ for each feature. `synth2` consists of two equal sized clusters sampled from two normal distributions $(\mu_0, \sigma_0^2) = (0, 1)$ and $(\mu_1, \sigma_1^2) = (2, 4)$ for each feature. `synth3` consists of three clusters with the size ratio $(2, 1, 1)$ sampled from three two-dimensional multivariate normal distributions, where the mean is randomly sampled from $[-25, 25]$ and the variance is always 1 for each feature. For each dataset, we obtained the averaged dendrogram purity from 10 trials.

We collected six real-world datasets from UCI machine learning repository (Lichman, 2013) and used only continuous features. The statistics of datasets are summarized in Table 1.

3.1 Results and Discussion

First we examine the sensitivity of our method with respect to the parameter k for k-nearest neighbor binarization using the synthetic dataset `synth1` with $n = 100$ and $m = 2$ and the real-world dataset `parkinsons` with $n = 197$ and $m = 23$. We plot results in Figure 2, where we varied k from 10 to 90 for `synth1` and from 10 to 190 for `parkinsons`. It shows that when $k \geq 20$, the dendrogram purity

[1] `http://research.nii.ac.jp/~uno/code/lcm53.zip`

Table 1: Experimental results, where c denotes the number of classes.

Name	n	m	c	# clusters		DP		Runtime (sec.)	
				Ours	HAC	Ours	HAC	Ours	HAC
`synth1`	100	2	2	77,364.5	199	**0.937**	0.812	1.24×10^{0}	1.01×10^{-3}
`synth1_large`	1,000	500	2	58.4	1,999	**1.0**	**1.0**	1.42×10^{1}	4.52×10^{-1}
`synth2`	100	2	2	27,445.3	199	**0.842**	0.705	5.88×10^{-1}	9.02×10^{-4}
`synth3`	100	2	3	425.2	199	**0.976**	0.936	1.64×10^{-1}	9.12×10^{-4}
`parkinsons`	197	23	2	263,189	393	**0.828**	0.738	1.30×10^{1}	5.32×10^{-3}
`vertebral`	310	6	2	503,476,064	619	**0.872**	0.686	2.55×10^{4}	4.25×10^{-3}
`breast_cancer`	569	10	2	3,142	1,137	**0.869**	0.771	3.40×10^{0}	8.43×10^{-3}
`wine_red`	1,600	12	2	24,412,834	3,199	**0.849**	0.845	3.73×10^{3}	8.09×10^{-2}
`ctg`	2,126	20	2	1,426,981	4,251	**0.800**	0.765	2.52×10^{2}	1.23×10^{-1}
`seismic_bumps`	2,584	25	2	91,059	5,167	0.931	**0.943**	9.48×10^{1}	1.52×10^{-1}

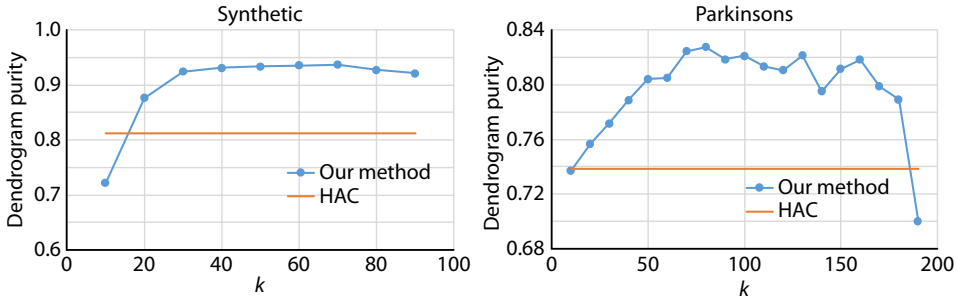

Figure 2: Result on `synth1` (left) and `parkinsons` (right).

is higher than HAC in both datasets and it is stable for larger k except for $k = 190$ in `parkinsons`, which is almost the same as the dataset size. This means that our method is robust to changes in k if k is set to be sufficiently large. In the following, we set k to be the half of the respective dataset size.

Next we examine the clustering performance of our method compared to HAC across various types of datasets. Results are summarized in Table 1. To prune unnecessary small clusters in our method, we set the lower bound of the size of clusters as 190 for `ctg`, `seismic_bumps`, and 490 for `synth1_large`. They clearly show that our method is consistently superior to HAC across all synthetic and real-world datasets except for `seismic_bumps`. The reason is that our method can learn overlapped clusters while HAC cannot. Although the number of clusters in HAC is always fixed to $2n - 1$ as it learns a binary tree, our method allows more flexible clustering, resulting in a larger number of clusters as shown in Table 1. How to effectively use the lower bound of the size of clusters to reduce clusters is our future work.

To summarize, our results show that the proposed method is robust to the parameter setting and can obtain better quality hierarchical structures than the standard baseline, hierarchical agglomerative clustering with Wald's method. This means that a graph representation learned by our method can be effective for further data analysis for relational reasoning.

4 Conclusions

In this paper, we have proposed a novel method that can learn graph structured representation of numerical data. Our method first binarizes a given dataset based on nearest neighbor search and then applies formal concept analysis (FCA) to the binarized data. The extracted concept lattice corresponds to a hierarchy of clusters, which leads to a directed graph representation. We have experimentally showed that our method can obtain more accurate hierarchical clusters compared to the standard hierarchical agglomerative clustering with Wald's method.

Acknowledgments: This work was supported by JSPS KAKENHI Grant Numbers JP16K16115, JP16H02870, and JST, PRESTO Grant Number JPMJPR1855, Japan (M.S.); and JSPS KAKENHI Grant Number 15H05711 (T.W.).

References

C. C. Aggarwal and J. Han, editors. *Frequent Pattern Mining*. Springer, 2014.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based *n*-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2 edition, 2002.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649. IEEE, 2013.

K. A. Heller and Z. Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 297–304, 2005.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL `http://www.scipy.org/`. [Online; accessed <today>].

M. Kaytoue, S. O. Kuznetsov, A. Napoli, and S. Duplessis. Mining gene expression data with pattern structures in formal concept analysis. *Information Sciences*, 181:1989–2001, 2011.

A. Kobren, N. Monath, A. Krishnamurthy, and A. McCallum. A hierarchical algorithm for extreme clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 255–264, 2017.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, 2012.

M. Lichman. UCI machine learning repository, 2013.

O. Maimon and L. Rokach, editors. *Data Mining and Knowledge Discovery Handbook*. Springer, 2005.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, 2013.

N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1):25–46, 1999.

A. Santoro, D. Raposo, D. G. T. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. *arXiv:1706.01427*, 2017.

T. Uno, T. Asai, Y. Uchida, and H. Arimura. An efficient algorithm for enumerating closed patterns in transaction databases. In *Discovery Science*, volume 3245 of *LNCS*, pages 16–31, 2004.

P. Valtchev, R. Missaoui, and R. Godin. Formal concept analysis for knowledge discovery and data mining: The new challenges. In *Concept Lattices*, volume 2961 of *LNCS*, pages 352–371, 2004.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

F. Zhou, F. D. l. Torre, and J. K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3): 582–596, 2013.