Constrained Clustering using Gaussian Processes

Panagiotis A. Traganitis, and Georgios B. Giannakis
Dept. of Electrical and Computer Engineering & Digital Technology Center, University of Minnesota, USA.

***Abstract*—Constrained clustering is an important machine learning, signal processing and data mining tool, for discovering clusters in data, in the presence of additional domain information. The present work introduces a probabilistic scheme for constrained clustering based on the popular Gaussian Process framework. The proposed scheme accommodates pairwise, must- and cannot-link constraints between data, does not require hyperparameter tuning, and enables assessment of the reliability of obtained results. Preliminary results on real data showcase the potential of the proposed approach.**

***Index Terms*—Constrained clustering, clustering, Gaussian process**

I. Introduction

Clustering (a.k.a. unsupervised classification), the task of assigning data to groups, in the absence of group labels, is a popular data analytic tool, used in multiple areas such as machine learning, data mining, and signal processing [1]–[3].

In many cases, additional domain knowledge may be available. Such domain knowledge is typically encoded in so-called must-link (ML) and cannot-link (CL) pairwise constraints [4]. Each of these constraints encodes the relationships between two data points and indicates whether they should or should not belong in the same cluster. Clustering with constraints finds numerous applications in areas such as handwritten character recognition [5], video surveillance [6], community detection and image segmentation [7], to name a few.

Multiple algorithms have been developed to tackle the constrained clustering task, using ML and CL constraints. A constrained version of the popular k-means algorithm was proposed in [8]. Other methods utilize spectral clustering approaches [9]. In [10], [11] only ML constraints are incorporated in the data similarity matrix used for spectral clustering, whereas [12] adapts the embedding obtained by spectral clustering to satisfy both ML and CL constraints, using semi-definite programming. The method of [13] also utilizes the data similarity matrix of spectral clustering, by incorporating both ML and CL constraints, and attempts to solve exactly 2-way spectral clustering. When more than 2 clusters are present, they are split recursively using 2-way spectral clustering. Recently, [7] introduced a spectral method that reduces constrained clustering into a generalized eigenvalue problem, resulting in an efficient algorithm.

The present work puts forth a novel probabilistic scheme for *constrained clustering* that leverages the popular and flexible Gaussian process framework, which has been utilized successfully for regression and classification [14]. By utilizing the co-association matrix representation of a clustering, we show that constrained clustering can be viewed as Gaussian process classification. Based on this, an algorithm to estimate clusters of data, that takes into account both ML and CL constraints, and automatically tunes all relevant hyperparameters, is developed. To the best of the authors knowledge, this work is the first to utilize Gaussian processes for constrained clustering.

Work in this paper was supported by NSF grants 1500713, 1514056, 1711471, and 1901134. Emails: {traga003,georgios}@umn.edu

Notation. Unless otherwise noted, lowercase bold letters, $\boldsymbol{x}$, denote column vectors, uppercase bold letters, $\mathbf{X}$, represent matrices, and calligraphic uppercase letters, $\mathcal{X}$, stand for sets. The (i,j)th entry of matrix $\mathbf{X}$ is denoted by $[\mathbf{X}]_{ij}$. Pr denotes probability, or the probability mass/density function(pmf/pdf); $\sim$ denotes "distributed as," and $\mathcal{N}(\boldsymbol{m},\mathbf{S})$ denotes the multivariate Gaussian distribution with mean $\boldsymbol{m}$ and covariance matrix $\mathbf{S}$.

II. Problem statement and preliminaries

Consider a dataset consisting of N data $\{\boldsymbol{x}_n\}_{n=1}^N$, each belonging to one of C possible clusters, and a $N\times 1$ vector $\boldsymbol{\pi}$ whose entries indicate the cluster each datum belongs to, that is $[\boldsymbol{\pi}]_n = 2$ if $\boldsymbol{x}_n$ belongs to the second cluster. Instead of $\boldsymbol{\pi}$, a clustering of N data can be represented using the so-called $N\times N$ co-association matrix $\mathbf{A}$, with entries $a_{n,n'} := [\mathbf{A}]_{n,n'} = 1$ if $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n'}$ belong to the same cluster and are 0 otherwise. Therefore, knowledge of $\mathbf{A}$ is equivalent to knowing the clusters of the data. The co-association matrix can also be thought of as the binary adjacency matrix of a graph $\mathcal{G}$, where every cluster corresponds to a fully connected component. Since $\mathbf{A}$ is symmetric, and all its diagonal entries are equal to 1 (a datum is in the same cluster with itself), specifying the $\bar{N} = \binom{N}{2}$ entries of its upper (or lower) triangular part is sufficient for clustering.

In addition to the data, must-link and cannot-link constraints are provided, and collected in the corresponding sets $\mathcal{C}_{\rm ML}$ and $\mathcal{C}_{\rm CL}$. All constraints consist of tuples (i,j) corresponding to entries of the co-association matrix $\mathbf{A}$, that is constraint (i,j) corresponds to $a_{i,j}$. A must-link constraint (i,j) indicates that two data points, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ must belong to the same cluster, i.e. $a_{i,j}=1$, whereas a cannot-link constraint (i',j') indicates that two points $\boldsymbol{x}_{i'}$ and $\boldsymbol{x}_{j'}$ are not in the same cluster, $a_{i',j'}=0$.

Let $\mathcal{C} = \mathcal{C}_{\rm ML}\cup\mathcal{C}_{\rm CL}$. The task of *constrained clustering* is given N data $\{\boldsymbol{x}_n\}_{n=1}^N$ and $N_c = |\mathcal{C}|$ constraints, to find the cluster each datum belongs to, or equivalently estimate the entries of the $N\times N$ matrix $\mathbf{A}$.

To perform clustering using the available constraints, we will utilize the Gaussian Process framework [14]. In particular, we will focus on learning a latent function $f:\mathbb{R}^D\times\mathbb{R}^D\rightarrow$

$\mathbb{R}$. This latent function captures the connectivity between any two data points in the dataset. Let $f_{n,n'} := f(\boldsymbol{x}_n, \boldsymbol{x}_{n'})$. The relationship between the latent function f and the entries $a_{n,n'}$ of the co-association matrix is assumed to be the following

$$\Pr(a_{n,n'} = 1 | f_{n,n'}) = \sigma(f_{n,n'}), \tag{1}$$

where $\sigma(x) = 1/(1+\exp(-x))$ denotes the so-called sigmoid function, which maps the value of $f_{n,n'}$ to the values in $[0, 1]$. Let $\boldsymbol{y}$ be a $N_c \times 1$ binary vector containing the values all provided constraints, i.e. $y_n := [\boldsymbol{y}]_n = a_{i_n,j_n}$, for $n = 1, \ldots, N_c$ and $(i_n, j_n) \in \mathcal{C}$. Also let $\boldsymbol{f} = [f_1, \ldots, f_{N_c}]^\top$ be a $N_c \times 1$ vector of corresponding latent variables. The conditional pmf of $\boldsymbol{y}$ given $\boldsymbol{f}$ is then

$$\Pr(\boldsymbol{y}|\boldsymbol{f}) = \prod_{n=1}^{N_c} \Pr(y_n|f_n) = \prod_{n=1}^{N_c} \sigma(f_n)^{y_n}(1-\sigma(f_n))^{1-y_n}$$

Next, we will assume that realizations of the latent function f follow a Gaussian Process model. This implies

$$\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, \mathbf{K}) \tag{2}$$

where $\mathbf{K}$ is a predefined covariance or kernel matrix, that encodes the relationships between values of the latent function. Contrary to kernels used typically in machine learning [15], this particular kernel does not directly capture the similarity between two data points $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n'}$. Instead, it quantifies the relationship between pairs of data, $\{\boldsymbol{x}_n, \boldsymbol{x}_{n'}\}$ and $\{\boldsymbol{x}_{n''}, \boldsymbol{x}_{n'''}\}$. In order to define it, consider first $\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'})$, a kernel function that captures the relationship between a pair of data $\boldsymbol{x}_n, \boldsymbol{x}_{n'}$. Examples of kernel functions include the linear kernel $\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \gamma \boldsymbol{x}_n^\top \boldsymbol{x}_{n'}$, or the squared exponential kernel $\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \gamma \exp\left(-\|\boldsymbol{x}_n - \boldsymbol{x}_{n'}\|_2^2/(2\nu^2)\right)$. Here, $\gamma, \nu > 0$ are tunable kernel parameters, collected in $\boldsymbol{\theta} = [\gamma, \nu]$. Using κ, a kernel (similarity) between entries $f_{n,n'}$ and $f_{n'',n'''}$ can then be computed as follows

$$\begin{aligned} K\left(\{n, n'\}, \{n'', n'''\}\right) = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n''})\kappa(\boldsymbol{x}_{n'}, \boldsymbol{x}_{n'''}) \\ +\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'''})\kappa(\boldsymbol{x}_{n'}, \boldsymbol{x}_{n''}). \end{aligned} \tag{3}$$

Such a kernel has been used successfully for link prediction and analysis in graphs [16]. With $(i_n, j_n) \in \mathcal{C}$ denoting the tuple corresponding to the n-th constraint, the $N_c \times N_c$ kernel matrix $\mathbf{K}$ has entries

$$[\mathbf{K}]_{n,n'} = K\left(\{i_n, j_n\}, \{i_{n'}, j_{n'}\}\right) \tag{4}$$

Note that, since κ depends on the tunable parameters $\boldsymbol{\theta}$, so does the kernel $\mathbf{K}$. Finally, the joint pdf of the latent variables $\boldsymbol{f}$ and provided constraints $\boldsymbol{y}$ is

$$\begin{aligned} \Pr(\boldsymbol{f}, \boldsymbol{y}) &= \Pr(\boldsymbol{y}|\boldsymbol{f}) \Pr(\boldsymbol{f}) \\ &= \left(\prod_{n=1}^{N_c} \Pr(y_n|f_n)\right) \Pr(\boldsymbol{f}) \end{aligned} \tag{5}$$

Having defined $\boldsymbol{y}, \boldsymbol{f}$ and $\mathbf{K}$, it can be seen that the constrained clustering task has been converted to that of Gaussian Process Classification, where the available constraints $\boldsymbol{y} \in \{0, 1\}^{N_c \times 1}$ act as training data. The next section introduces the proposed algorithm for constrained clustering.

III. Gaussian Process Constrained Clustering

Given a set of N_c must- and cannot-link constraints, vectorized in $\boldsymbol{y} \in \{0, 1\}^{N_c \times 1}$, and their corresponding data points $\{\boldsymbol{x}_n\}$, we would like to estimate the entries of the co-association matrix $\mathbf{A}$, and consequently the C clusters of data. Similar to Gaussian process regression and classification, in order to predict the remaining entries of $\mathbf{A}$, knowledge of the posterior $\Pr(\boldsymbol{f}|\boldsymbol{y}) = \frac{\Pr(\boldsymbol{f}, \boldsymbol{y})}{\Pr(\boldsymbol{y})}$ is crucial. Due to the presence of the sigmoid functions in $\Pr(\boldsymbol{y}|\boldsymbol{f})$ directly computing the marginal

$$\Pr(\boldsymbol{y}) = \int \Pr(\boldsymbol{y}, \boldsymbol{f}) d\boldsymbol{f} = \int \Pr(\boldsymbol{y}|\boldsymbol{f}) \Pr(\boldsymbol{f}) d\boldsymbol{f} \tag{6}$$

is intractable. To overcome this issue, Markov chain monte carlo or Expectation propagation approaches can be utilized [14], however, here we opted for the following variational lower bound of the sigmoid [17], which has been successfully used for Gaussian Process Classification in [18], [19]: For any $\xi > 0$ it holds

$$\sigma(x) = \frac{1}{1+e^{-x}} \geq \sigma(\xi) \exp\left(\frac{x-\xi}{2} - \lambda(\xi)(x^2 - \xi^2)\right) \tag{7}$$

where $\lambda(\xi) = (\sigma(\xi) - 1/2)/(2\xi)$. Applying (7) to all terms of $\Pr(\boldsymbol{y}|\boldsymbol{f})$, it is straightforward to show that

$$\begin{aligned} &\Pr(\boldsymbol{y}|\boldsymbol{f}) \geq g(\boldsymbol{\xi}, \boldsymbol{f}, \boldsymbol{y}) := \\ &\prod_{n=1}^{N_c} \sigma(\xi_n) \exp\left(f_n(y_n - \frac{1}{2}) - \lambda(\xi_n)(f_n^2 - \xi_n^2) - \frac{\xi_n}{2}\right) \end{aligned} \tag{8}$$

where $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_{N_c}]^\top$ collects the N_c newly introduced ξ variables. As $g(\boldsymbol{\xi}, \boldsymbol{f}, \boldsymbol{y})$ is a quadratic function of $\boldsymbol{f}$, a lower bound approximation to the marginal (6) can be computed, namely

$$q(\boldsymbol{y}) := \int g(\boldsymbol{\xi}, \boldsymbol{f}, \boldsymbol{y}) \Pr(\boldsymbol{f}) d\boldsymbol{f}.$$

Since $\Pr(\boldsymbol{f}|\boldsymbol{y}) \propto \Pr(\boldsymbol{y}|\boldsymbol{f}) \Pr(\boldsymbol{f})$, we can also compute an approximate posterior

$$q(\boldsymbol{f}|\boldsymbol{y}) \propto g(\boldsymbol{\xi}, \boldsymbol{f}, \boldsymbol{y}) \Pr(\boldsymbol{f}) \propto \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{9}$$

where $\boldsymbol{\Sigma} = \left(\mathbf{K}^{-1} + 2\boldsymbol{\Lambda}\right)^{-1}$, $\boldsymbol{\Lambda}$ is a diagonal matrix with $\boldsymbol{\lambda} = [\lambda(\xi_1), \ldots, \lambda(\xi_{N_c})]$ in its diagonal, and $\boldsymbol{\mu} = \boldsymbol{\Sigma}(\boldsymbol{y} - \frac{1}{2}\mathbf{1})$, with $\mathbf{1}$ denoting the all ones vector of appropriate dimension. Note that, use of the bound in (7) requires estimation of the variables $\boldsymbol{\xi}$.

Kernel parameters $\boldsymbol{\theta}$ can be estimated by maximizing the approximate marginal $q(\boldsymbol{y})$, or equivalently its logarithm, i.e.

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ q(\boldsymbol{y}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log \int g(\boldsymbol{\xi}, \boldsymbol{f}, \boldsymbol{y}) \Pr(\boldsymbol{f}) d\boldsymbol{f} \tag{10}$$

In this case, (10) boils down to the following optimization problem

$$\min_{\boldsymbol{\theta}} \quad \log|\mathbf{K} + \frac{1}{2}\boldsymbol{\Lambda}^{-1}| + \frac{1}{4}\bar{\boldsymbol{y}}^\top \left(\mathbf{K} + \frac{1}{2}\boldsymbol{\Lambda}^{-1}\right)^{-1} \bar{\boldsymbol{y}} \tag{11}$$

where $\bar{\boldsymbol{y}} = \boldsymbol{\Lambda}^{-1}(\boldsymbol{y} - \frac{1}{2}\mathbf{1})$ and even though the optimization in (11) is nonconvex, it can be carried out using conjugate

gradient methods, as is common for GP models [14]. Similarly, $\boldsymbol{\xi}$ is estimated by maximizing $\log q(\boldsymbol{y})$, and entries of $\boldsymbol{\xi}$ are updated as

$$\frac{\partial \log q(\boldsymbol{y})}{\partial \xi_n} = 0 \Rightarrow \hat{\xi}_n = \sqrt{[\boldsymbol{\mu}]_n^2 + [\boldsymbol{\Sigma}]_{nn}}. \quad (12)$$

The parameter updates of (10), (12) and evaluation of the posterior (9) are carried out in an alternating fashion, until convergence. The computational complexity of this part of the constrained clustering algorithm is $\mathcal{O}(IN_c^3)$, where I is the number of iterations until convergence.

A. Estimating entries of the co-association matrix

Having learned the kernel parameters, and obtained the approximate posterior $q(\boldsymbol{f}|\boldsymbol{y})$ using the provided constraints, we now have to estimate the remaining $\bar{N}_c = \bar{N} - N_c$ entries of the unknown co-association matrix $\mathbf{A}$. Let $\mathcal{A}$ be a set containing the tuples corresponding to the unknown entries of $\mathbf{A}$. Using the approximate posterior $q(\boldsymbol{f}|\boldsymbol{y})$, the conditional pmf of the (n,n')-th entry of $\mathbf{A}$, $(n,n') \in \mathcal{A}$ is

$$\Pr(a_{n,n'}|\boldsymbol{y}) = \int \Pr(a_{n,n'}, f_{n,n'}, \boldsymbol{f}|\boldsymbol{y}) df_{n,n'} d\boldsymbol{f} \approx$$
$$\int \Pr(a_{n,n'}|f_{n,n'}) \left(\int \Pr(f_{n,n'}|\boldsymbol{f}) q(\boldsymbol{f}|\boldsymbol{y}) d\boldsymbol{f} \right) df_* \quad (13)$$

where $f_{n,n'}$ is the unknown latent variable corresponding to $a_{n,n'}$. As the latent variables follow a Gaussian distribution, the conditional distribution of $f_{n,n'}$ given $\boldsymbol{f}$, is also a Gaussian

$$\Pr(f_{n,n'}|\boldsymbol{f}) = \mathcal{N}(\boldsymbol{k}_{n,n'}^\top \mathbf{K}^{-1} \boldsymbol{f}, k_{nn'} - \boldsymbol{k}_{n,n'}^\top \mathbf{K}^{-1} \boldsymbol{k}_{n,n'}) \quad (14)$$

where $\boldsymbol{k}_{n,n'} = [K(\{n,n'\},\{i_1,j_2\}), .., K(\{n,n'\},\{i_{N_c},j_{N_c}\})]$, $(i_l, j_l) \in \mathcal{C}$ denotes the tuple corresponding to the l-th constraint, and $k_{nn'} = K(\{n,n'\},\{n,n'\})$ [cf. (3)] [14]. Then

$$\int \Pr(f_{n,n'}|\boldsymbol{f}) q(\boldsymbol{f}|\boldsymbol{y}) d\boldsymbol{f} = \mathcal{N}(m, s^2) \quad (15)$$

with $m = \boldsymbol{k}_{n,n'}^\top \mathbf{K}^{-1} \boldsymbol{\mu}$ and $s^2 = k_{nn'} - \boldsymbol{k}_{n,n'}^\top \left(\mathbf{K} + \frac{1}{2}\boldsymbol{\Lambda}^{-1}\right)^{-1} \boldsymbol{k}_{n,n'}$ [cf. (9)]. Using (15) alongside the results from [17, Chapter 4.5.2] we have

$$\Pr(a_{n,n'} = 1|\boldsymbol{y}) \approx \sigma(\rho(s^2)m) \quad (16)$$

with $\rho(s^2) = (1 + \pi s^2/8)^{-1/2}$. Finally, if $\Pr(a_{n,n'} = 1|\boldsymbol{y}) \geq 1/2$, we set $\hat{a}_{n,n'} = [\hat{\mathbf{A}}]_{n,n'} = 1$ and 0 otherwise. The computational complexity of evaluating the unknown entries of $\mathbf{A}$ is $\mathcal{O}(\bar{N}_c N_c^2)$.

Remark 1: The variance s^2 in (15) indicates how uncertain our model is with respect to the $f_{n,n'}$ it has estimated. Such uncertainty quantification may be especially useful in active learning setups [20], [21].

B. The constrained clustering algorithm

The entire constrained clustering algorithm is listed in Alg. 1. Given a set of must- and cannot-link constraints in $\boldsymbol{y}$, kernel parameters $\boldsymbol{\theta}$ and an approximate posterior $q(\boldsymbol{f}|\boldsymbol{y})$ are estimated. The estimated parameters and posterior are then used to estimate the entries of $\mathbf{A}$, as outlined in the previous subsection. The estimated entries along with the provided constraints form the estimated co-association matrix $\hat{\mathbf{A}}$, which indicates the clustering result. In many cases, $\hat{\mathbf{A}}$ might be noisy. Therefore, we obtain final clustering indicator vector $\hat{\boldsymbol{\pi}}$ using spectral clustering [9] with $\hat{\mathbf{A}}$ as the adjacency matrix.

Algorithm 1 Constrained Clustering using Gaussian Processes

Input: Constraints $\boldsymbol{y}$; Kernel $\mathbf{K}$; initial parameters $\boldsymbol{\theta}^{(0)}$.
Output: Estimates $\hat{\boldsymbol{\theta}}$; $\hat{\mathbf{A}}$; $\hat{\boldsymbol{\pi}}$; posterior $q(\boldsymbol{f}|\boldsymbol{y})$

```
while not converged do
    Compute θ̂ using (11).
    Evaluate posterior q(f|y) using (9).
    Compute ξ̂ using (12).
end while
for (n, n') ∈ A do
    Compute Pr(a_{n,n'} = 1|y) using (15), (16)
    If Pr(a_{n,n'} = 1|y) ≥ 1/2 set [Â]_{n,n'} = 1 and 0 otherwise.
end for
Run Spectral Clustering using estimated Â and obtain π̂
```

The next section will evaluate the performance of the proposed constrained clustering scheme.

IV. Numerical tests

The performance of the proposed constrained clustering method of Alg. 1 (denoted as *CCGP*) is evaluated using real datasets. *CCGP* is compared to spectral clustering [9], that does not account for the available constraints, denoted as *SC*, and the state-of-the-art methods of [7] and [13], denoted as *FAST-GE* and *COSC* respectively. The metric used to evaluate clustering performance is Normalized Mutual Information (NMI), defined as

$$\text{NMI} = \frac{I(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}})}{H(\boldsymbol{\pi}) + H(\hat{\boldsymbol{\pi}})}$$

with $\boldsymbol{\pi}$ being the ideal cluster indicator vector, $\hat{\boldsymbol{\pi}}$ the estimated one, $I(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}})$ denoting the mutual information between $\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}$ and $H(\boldsymbol{\pi})$ denoting the Shannon entropy of $\boldsymbol{\pi}$ [22]. All results represent the average of 10 independent Monte Carlo runs, and for all datasets considered, in each run, N_c constraints are randomly generated and provided to the constrained clustering algorithms. In all tests, the base kernel [cf. (3)] used is the squared exponential one $\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \gamma \exp\left(-\|\boldsymbol{x}_n - \boldsymbol{x}_{n'}\|_2^2/(2\nu^2)\right)$ and the parameters $\boldsymbol{\theta} = [\gamma, \nu]$ are tuned automatically by Alg. 1. *SC* uses the squared exponential kernel with $\gamma = 1$ and ν^2 estimated as the average squared euclidean distance between all data.

Three real datasets from the UCI database [23] are considered, namely the Iris, Wine and Ionosphere datasets, as well as the Extended Yale Face database B [24]. The Iris dataset contains $N = 150$ data of size $D = 4$ belonging to $C = 3$ clusters, whereas the wine dataset contains $N = 178$ data vectors of size $D = 13$, organized into $C = 3$ clusters. The ionosphere dataset consists of $N = 351$ data vectors of

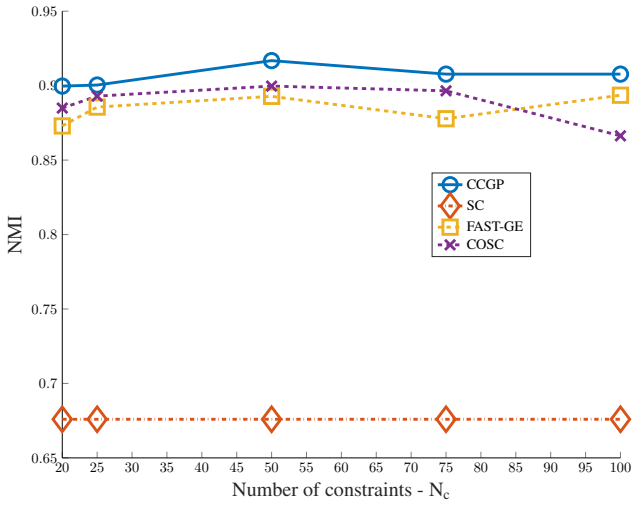

Fig. 1. Results for the Iris dataset.

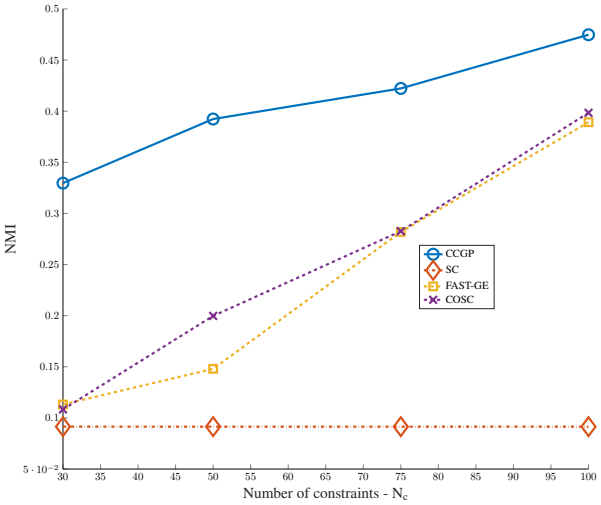

Fig. 3. Results for the Wine dataset.

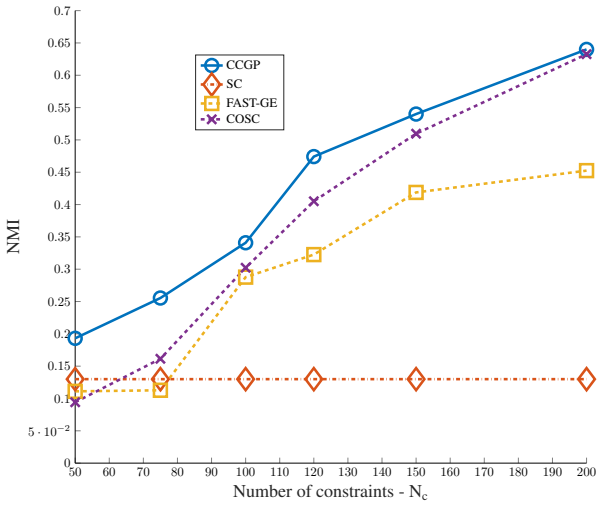

Fig. 2. Results for the Ionosphere dataset.

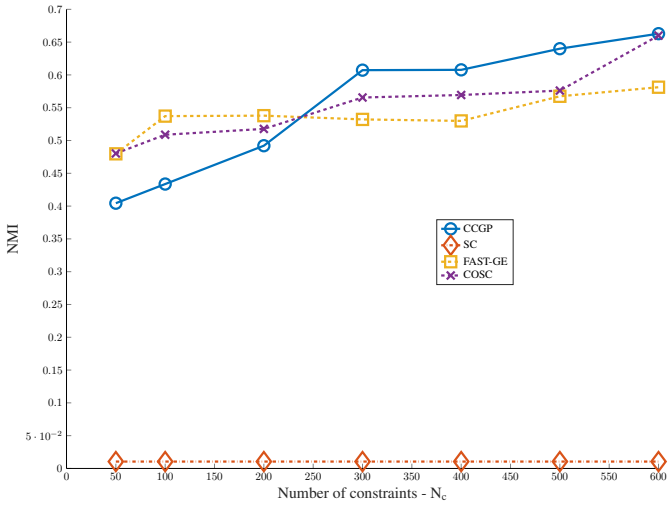

Fig. 4. Results for the subset of the Extended Yale dataset B.

dimension $D = 34$, belonging to $C = 2$ clusters. Furthermore, we are using a subset of the Extended Yale face database, with $N = 2,392$ image data of size $D = 1,024$, organized into $C = 3$ clusters.

Results for the Iris dataset, as the number of constraints N_c increases are shown in Fig. 1. For this dataset all constrained clustering methods significantly outperform SC, that does not take constraints into account. The proposed *CCGP* exhibits better NMI compared to both *Fast-GE* and *COSC*. A similar trend is observed for the Ionosphere and wine datasets in Figs. 2 and 3 respectively. For these datasets, as the number of constraints increases so does the performance of all constrained clustering algorithms. Fig. 4 shows the results for the subset of the Extended Yale Face database. Here, *SC* exhibits poor performance, suggesting that the squared exponential kernel may not be appropriate, however including constraints improves clustering performance dramatically. *CCGP* eventually outperforms *COSC* and *FAST-GE* as the number of constraints increases.

V. Conclusions and Future Research

This paper introduced a novel constrained clustering approach that utilizes the Gaussian Process framework. Preliminary tests on real data showcase the potential of the novel approach. Future research will focus on extensive numerical tests with real datasets, performance analysis, efficient algorithms that can handle large numbers of constraints, as well as online and active methods for constrained clustering.

References

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer, 2001.

[2] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 4th ed. Academic Press, 2008.

[3] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. John Wiley & Sons, 2001.

[4] S. Basu, I. Davidson, and K. Wagstaff, *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, 1st ed. Chapman & Hall/CRC, 2008.

[5] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. ICML '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 11. [Online]. Available: https://doi.org/10.1145/1015330.1015360

[6] Rong Yan, Jian Zhang, Jie Yang, and A. G. Hauptmann, "A discriminative learning framework with pairwise constraints for video object classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 578–593, April 2006.

[7] M. Cucuringu, I. Koutis, S. Chawla, G. Miller, and R. Peng, "Simple and scalable constrained clustering: a generalized spectral method," in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Gretton and C. C. Robert, Eds., vol. 51. Cadiz, Spain: PMLR, 09–11 May 2016, pp. 445–454.

[8] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, "Constrained k-means clustering with background knowledge," in *In ICML*. Morgan Kaufmann, 2001, pp. 577–584.

[9] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[10] S. D. Kamvar, D. Klein, and C. D. Manning, "Spectral learning," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, ser. IJCAI'03. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, p. 561–566.

[11] Zhengdong Lu and M. A. Carreira-Perpinan, "Constrained spectral clustering through affinity propagation," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, June 2008, pp. 1–8.

[12] Z. Li, J. Liu, and X. Tang, "Constrained clustering via spectral regularization," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 421–428.

[13] S. S. Rangapuram and M. Hein, "Constrained 1-spectral clustering," in *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, N. D. Lawrence and M. Girolami, Eds., vol. 22. La Palma, Canary Islands: PMLR, 21–23 Apr 2012, pp. 1143–1151. [Online]. Available: http://proceedings.mlr.press/v22/sundar12.html

[14] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[15] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[16] K. Yu and W. Chu, "Gaussian process models for link analysis and transfer learning," in *Advances in Neural Information Processing Systems 20*, 2008, pp. 1657–1664.

[17] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[18] P. Ruiz, P. Morales-Álvarez, R. Molina, and A. K. Katsaggelos, "Learning from crowds with variational gaussian processes," *Pattern Recognition*, vol. 88, pp. 298 – 311, 2019.

[19] P. Morales-Álvarez, A. Pérez-Suay, R. Molina, and G. Camps-Valls, "Remote sensing image classification with large-scale gaussian processes," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 2, pp. 1103–1114, Feb 2018.

[20] B. Settles, "Active learning," *Synthesis Lectures on Artif. Intel. and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[21] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *J. of Knowl. and Infor. Systems*, vol. 35, no. 2, pp. 249–283, 2013.

[22] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. USA: Wiley-Interscience, 2006.

[23] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[24] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 23, no. 6, pp. 643–660, June 2001.