Incremental Constrained Clustering by Minimal Weighted Modification

仝佳驹

2024-12-10

1 Background

2 Framework

3 Modification

4 Constraint Optimization

5 Active Constraint Selection

6 Experiments

7 conclusion

Clustering

- Clustering
  给定数据集 $X = \{x_i\}_{i=1}^{n}$，将数据集划分为若干个不相交的子集，每个子集称为一个簇。同一簇中的数据对象之间相似度较高，不同簇中的数据对象之间相似度较低。
  一种无监督学习
- Constraint Clustering
  约束聚类旨在通过约束的形式找到满足某些性质的聚类结果。
  例如：ML(Must-Link) 约束，CL(Cannot-Link) 约束。
  将无监督学习转化为半监督学习。
- Incremental Constraint Clustering
  增量式约束聚类是指在已有的聚类结果上，通过添加约束，来调整原有的聚类结果。

Human in the Loop

在实践中需要领域专家的知识来指导聚类过程
困难：只基于数据给出约束。
简易：根据当前聚类结果提供反馈来改进聚类。

- Requests：与人交互，因此每步的更新应够快。与前一步的结果相似。
- Problems：如何整合约束信息，如何保持聚类结果的稳定性，如何处理相矛盾的约束。

1 Background

2 Framework

3 Modification

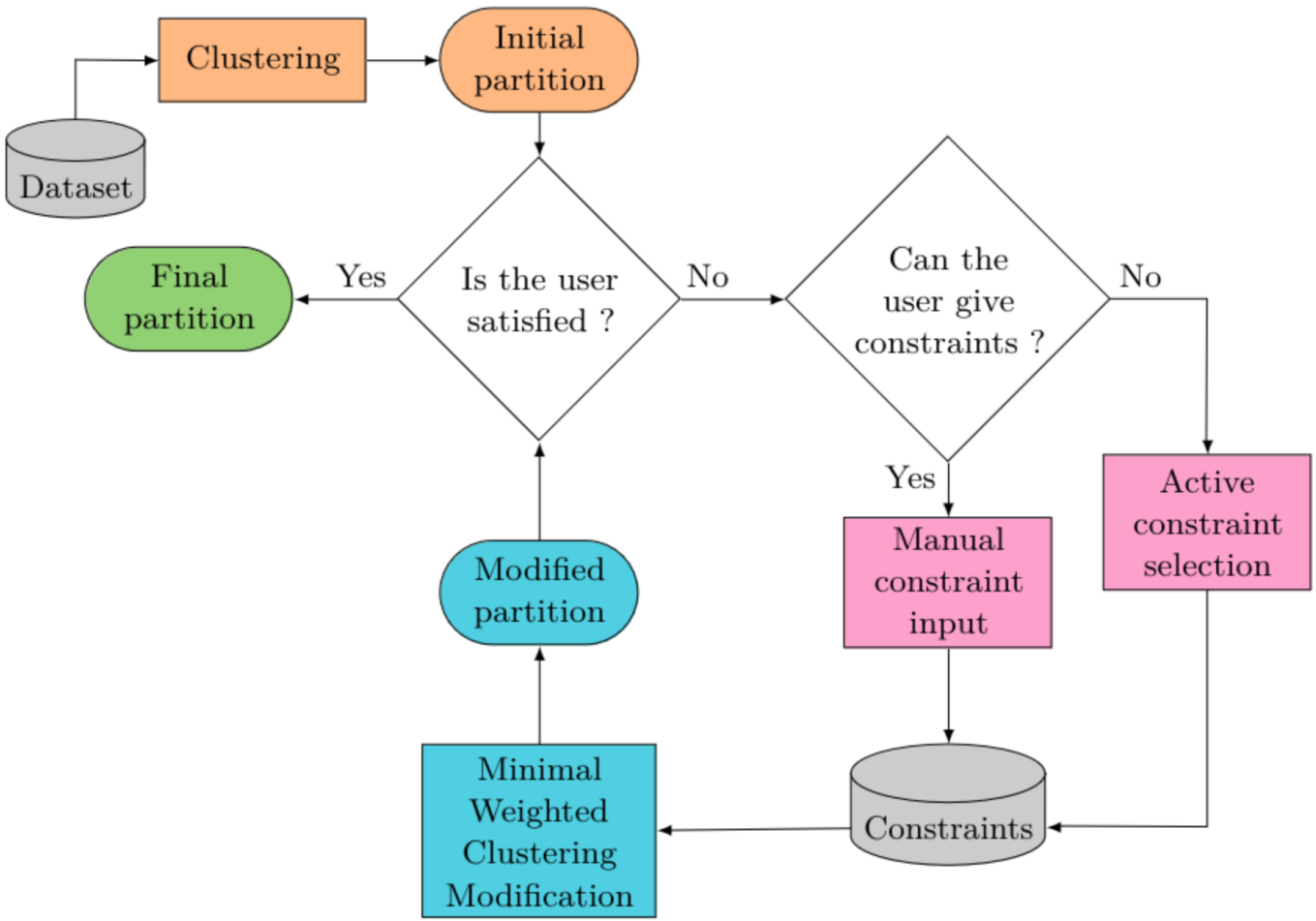
4 Constraint Optimization

5 Active Constraint Selection

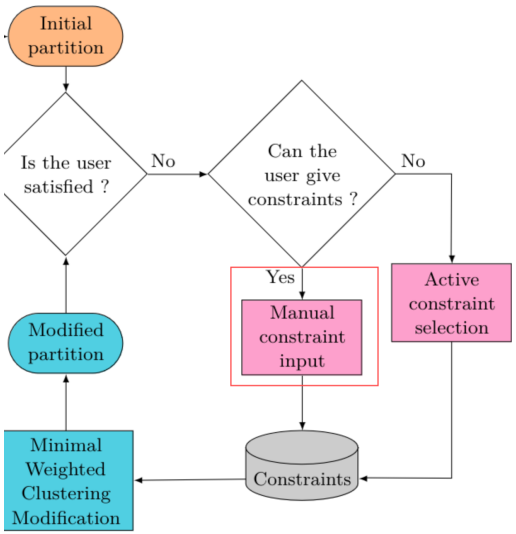
6 Experiments

7 conclusion

Incremental and Active Clustering Framework



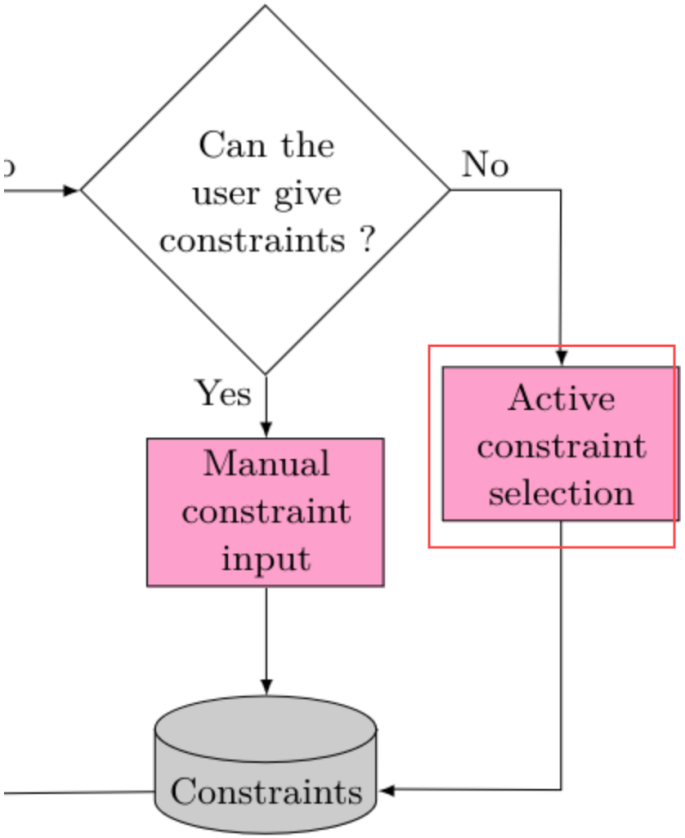
Mannual Constraints



用户给出约束

- Must-Link&Cannot-Link
- Triplets Constraints:(x, p, n)
  x 与 p 比 n 更相似。即
  $G_x = G_n \Rightarrow G_x = G_p$
- Span-limited
  Constraints:$S \subset X, C \subseteq [1, K]$
  S 中的点只能在 C 中的簇中。
- a generic span-limited
  constraint:given γ
  S 中的点最多在 γ 个簇中。
- implicit constraint:
  $P \Rightarrow Q$， P,Q 为 ML/CL 的合取式

Active Constraints Selection

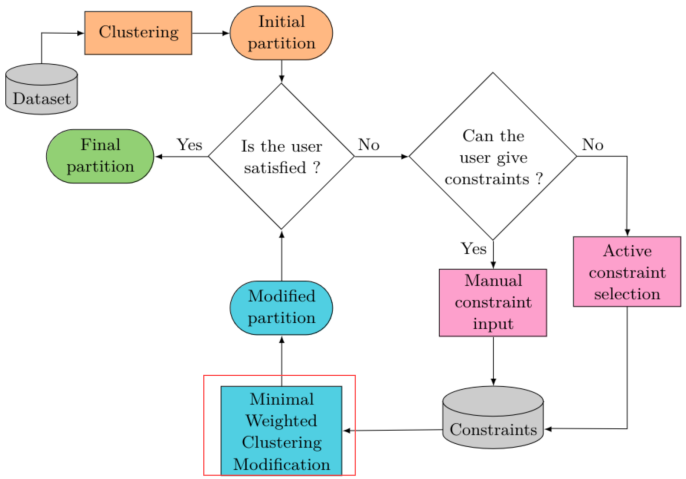


选择 most informative 的点，query 它与其他数据间的约束。

具体方法：

NPU(Neighborhood-based Pairwise Uncertainty)

Modication



given partition $\mathcal{P}$,user constraints $\mathcal{C}$,...

根据 $\mathcal{C}$，对 $\mathcal{P}$ 进行修改。

f 为度量 partition 间的差异的函数。

$$\mathcal{P}' = \arg\min f(\mathcal{P}', \mathcal{P})$$

1 Background

2 Framework

3 Modification

4 Constraint Optimization

5 Active Constraint Selection

6 Experiments

7 conclusion

Minimal Weighted Clustering Modification(MWCM)

Algorithm 1 Minimal Weighted Clustering Modification.

Input: Dataset $\mathcal{X}$, partition $\mathcal{P}$, constraints $\mathcal{C}$, anchor generation rate α, super-instance rate β, constraint satisfaction rate δ

Output: modified partition $\mathcal{P}'$

1: $anchors \leftarrow$ ComputeRepresentatives$(\mathcal{X}, \mathcal{P}, \alpha)$
2: $X \leftarrow$ ComputeCOPInstances$(\mathcal{X}, \mathcal{P}, \mathcal{C}, \beta)$
3: $\mathcal{D} \leftarrow$ DistanceMatrix$(X, anchors)$
4: $p \leftarrow$ GetConstrainedPartition$(X, \mathcal{P})$
5: $mods \leftarrow$ SolveModel$(\mathcal{D}, p, \mathcal{C}, \delta)$
6: **return** ApplyModifications$(mods, \mathcal{P})$

图 1: MWCM

1. 计算 $\mathcal{P}$ 的代表: anchors
2. 计算 $\mathcal{P}$ 的 super instances(用于后续 Constraint Optimization Problem)
3. 创建距离矩阵 $\mathcal{D}$(用于后续 COP)
4. 计算 X 的 clustering p
5. 根据 $\mathcal{D}, p, \mathcal{C}$ 求解 X 新的 clustering $mods$
6. 从 $mods$ 得到新的 partition $\mathcal{P}'$

Objective Function

- naive objective function:

$$\arg\min \sum_{i=1}^{N} \mathbb{I}(\mathcal{P}[i] \neq \mathcal{P}'[i])$$

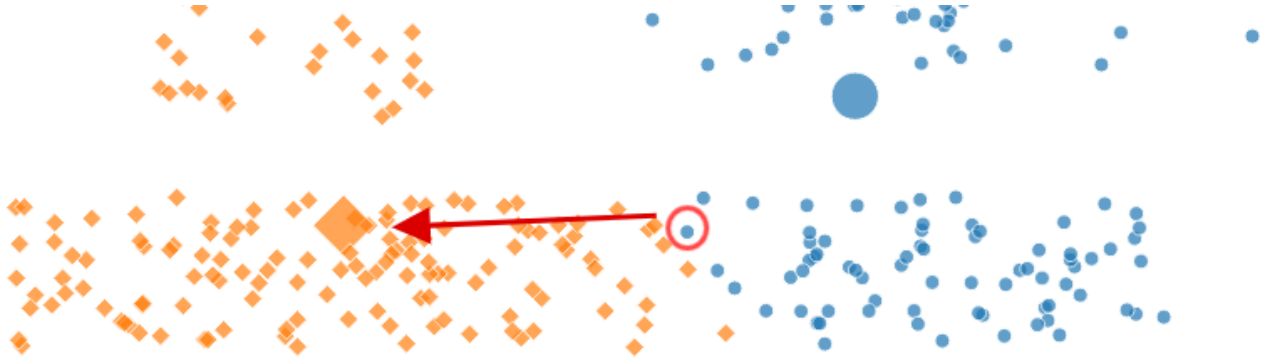
- take into account the structure of the clusters:

$$\arg\min \sum_{i=1}^{N} \mathbb{I}(\mathcal{P}[i] \neq \mathcal{P}'[i])\mathcal{D}[i, \mathcal{P}'[i]]$$

$\mathcal{D}[i, c]$ 为 i 与 cluster c 的距离。

Anchors

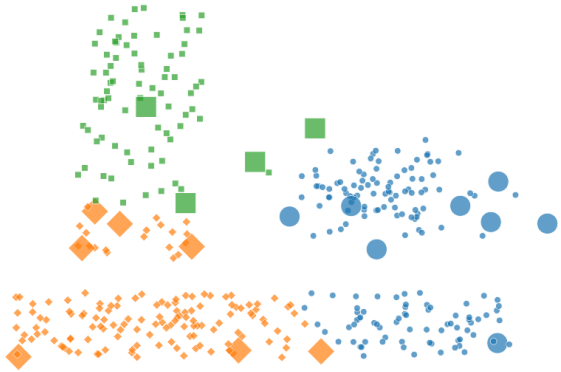
- $\mathcal{D}[i, c] = d(i, \mu_c)$,μ_c is cluster c's medoids. implicitly treat all clusters as spherical



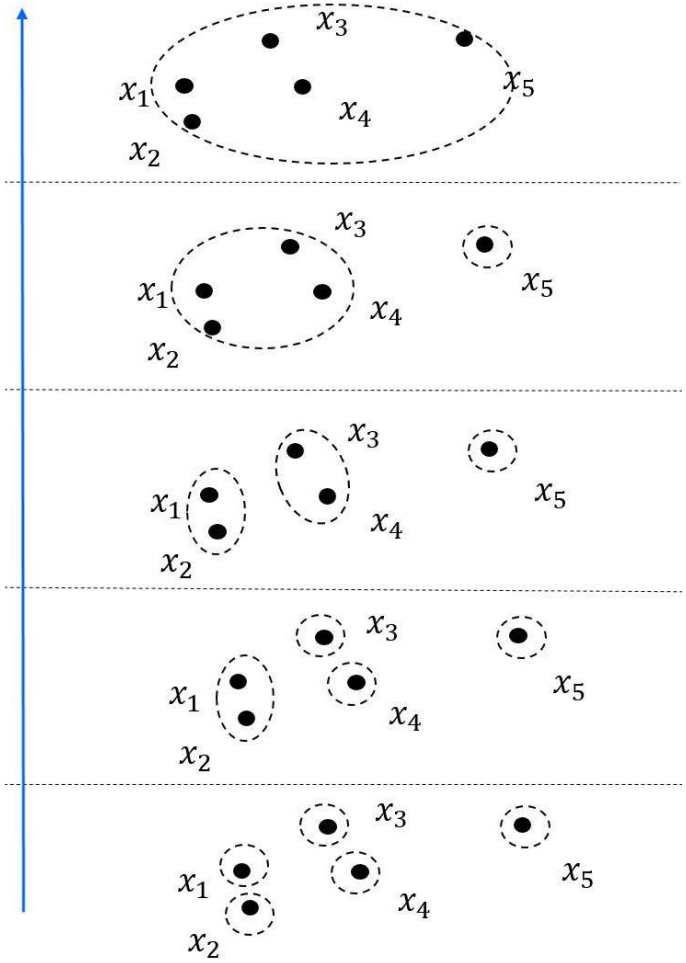
- anchors: representative points of sub-clusters of a cluster. $\mathcal{D}[i, c]$ represents the distance of instance i to its closest anchor belonging to cluster c

Anchors

- How to get sub-clusters of a cluster?⇐
  sigle-link hierarchical clustering
  适合发现任意形状的簇



- α:proportion of anchors per cluster.



Super Instances

Algorithm 1 Minimal Weighted Clustering Modification.

Input: Dataset $\mathcal{X}$, partition $\mathcal{P}$, constraints $\mathcal{C}$, anchor generation rate α, super-instance rate β, constraint satisfaction rate δ

Output: modified partition $\mathcal{P}'$

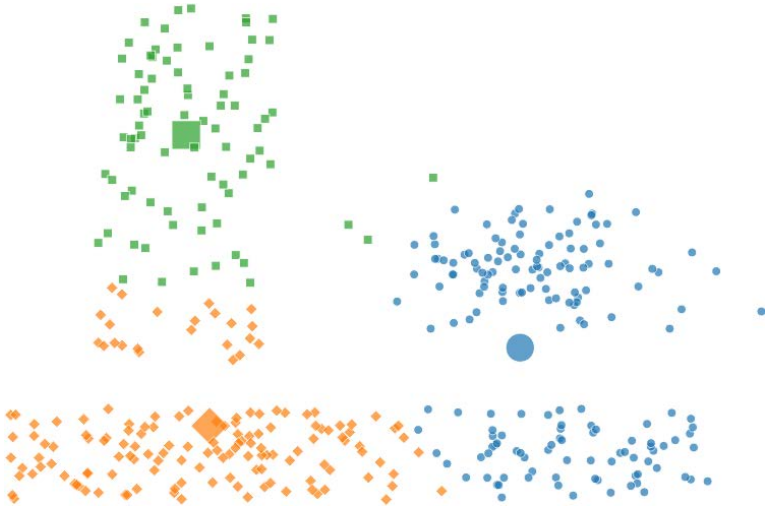
1: $anchors \leftarrow$ ComputeRepresentatives$(\mathcal{X}, \mathcal{P}, \alpha)$
2: $X \leftarrow$ ComputeCOPInstances$(\mathcal{X}, \mathcal{P}, \mathcal{C}, \beta)$
3: $\mathcal{D} \leftarrow$ DistanceMatrix$(X, anchors)$
4: $p \leftarrow$ GetConstrainedPartition$(X, \mathcal{P})$
5: $mods \leftarrow$ SolveModel$(\mathcal{D}, p, \mathcal{C}, \delta)$
6: **return** ApplyModifications$(mods, \mathcal{P})$

- 在实践中，专家只能对少数数据给出约束反馈，相对于数据集甚至可以忽略。
- 假设专家想要调整的是选择的点周围的区域内所有的点，而不是单单是点本身。
- super-instances:virtual instances grouping several real data points
- 推广被约束点的约束到其周围的点。

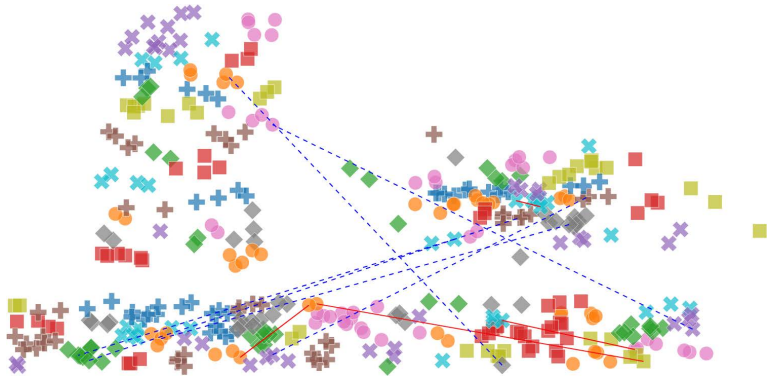
Super Instances

- How to get super instances?
  complete-link hierarchical clustering
- user constraints$\rightarrow$ super instances constraints
  潜在的约束冲突风险？
  确保每个 super instance **只有不多于一个被约束的数据点。**

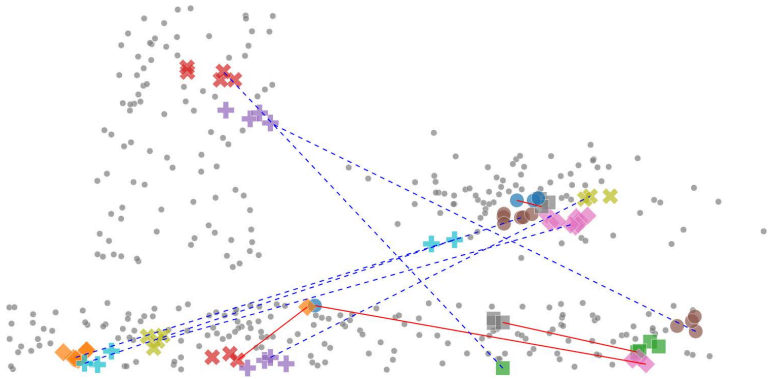
Super Instances



(a) previous partition



(b) complete-link clustering



(c) super instances

Apply Modification

Algorithm 1 Minimal Weighted Clustering Modification.

Input: Dataset $\mathcal{X}$, partition $\mathcal{P}$, constraints $\mathcal{C}$, anchor generation rate α, super-instance rate β, constraint satisfaction rate δ

Output: modified partition $\mathcal{P}'$

1: $anchors \leftarrow$ ComputeRepresentatives$(\mathcal{X}, \mathcal{P}, \alpha)$
2: $X \leftarrow$ ComputeCOPInstances$(\mathcal{X}, \mathcal{P}, \mathcal{C}, \beta)$
3: $\mathcal{D} \leftarrow$ DistanceMatrix$(X, anchors)$
4: $p \leftarrow$ GetConstrainedPartition$(X, \mathcal{P})$
5: $mods \leftarrow$ SolveModel$(\mathcal{D}, p, \mathcal{C}, \delta)$
6: **return** ApplyModifications$(mods, \mathcal{P})$

Algorithm 4 ApplyModifications.

Input : dataset $\mathcal{X}$, super-instances S, modifications $\mathcal{M}$, partition $\mathcal{P}$

Output : modified partition $\mathcal{P}'$

1: $\mathcal{P}' \leftarrow \mathcal{P}$
2: **for** each $sp \in S$ **do**
3: $\quad points \leftarrow \{x \in \mathcal{X} \mid x \in sp\}$
4: $\quad$ **for** each $p \in points$ **do**
5: $\quad\quad$ Update the membership of p in $\mathcal{P}'$ with the corresponding value in $\mathcal{M}$
return $\mathcal{P}'$

1 Background

2 Framework

3 Modification

4 Constraint Optimization

5 Active Constraint Selection

6 Experiments

7 conclusion

Variables

- variables:for each $i \in X, G_i$ with domain $[1, K]$
  $G_i = c$ means instance i to cluster c in new partition $\mathcal{P}'$
- objective function:

$$\arg\min \sum_{i \in X} \mathbb{I}(G_i \neq \mathcal{P}[i]) \mathcal{D}[i, G_i]$$

Handling Conflicting Constraints

- 可以设置 G_i 的 domain 为 $[1, K'], K' > K$，允许新的簇的产生。
  防止 instance 从优化上被分到新的簇：
  set $\mathcal{D}[i, k'], k \in [K+1, K']$ greater
- Relaxing constraints:
  $\sum_{c \in \mathcal{C}} S_c \overset{\geq}{=} \delta|\mathcal{C}|$, where $S_c = 1$ iff c is satisfied

1 Background

2 Framework

3 Modification

4 Constraint Optimization

5 Active Constraint Selection

6 Experiments

7 conclusion

Active Constraints Selection

- Neighborhoods $\mathcal{N}$:groups of instances whose cluster assignment is certain
- to construct $\mathcal{N}$,iteratively do:
  - select the most informative instance x^*
  - query the relationship between x^* N in $\mathcal{N}$,add ML/CL constraints
  - if no ML added,create a new neighborhood with x^*

Algorithm 2 Incremental NPU.

Input : Dataset $\mathcal{D}$, partition $\mathcal{P}$, oracle
Output : constraint set $\mathcal{C}$

$\mathcal{C} \leftarrow \emptyset$; $l \leftarrow 1$; $\mathcal{N} \leftarrow N_1 \mid N_1 = \{random(\mathcal{D})\}$
$x^* \leftarrow MostInformative(\mathcal{D}, \mathcal{P}, \mathcal{N})$
for each $N_i \in \mathcal{N}$ in decreasing order of $P(x^* \in N_i)$ **do**
    Query x^* against any $x_i \in N_i$ to the oracle
    if (x^*, x_i, ML) **then**
        $\mathcal{C} \leftarrow (x^*, x_i, ML)$
        $N_i = N_i \cup x^*$
        break
    else
        $\mathcal{C} \leftarrow (x^*, x_i, CL)$
if no ML is returned **then**
    $l++$; $N_l = x^*$; $\mathcal{N} \leftarrow \mathcal{N} \cup N_l$
return $\mathcal{C}$

Active Constraints Selection

- Neighborhood-based Pairwise Uncertainty(NPU)
  - $H(\mathcal{N}|x)$:entropy measure of the uncertainty to assign x to a neighborhood in $\mathcal{N}$
  - $\mathbb{E}(q(x))$:expected number of queries to discover its neighborhood
  - informativeness: $\frac{H(\mathcal{N}|x)}{\mathbb{E}(q(x))}$

Algorithm 3 MostInformative.

Input : Dataset $\mathcal{D}$, partition $\mathcal{P}$, set of neighborhoods $\mathcal{N}$
Output : most informative data point x^*

Learn a random forest classifier using $\mathcal{P}$ as labels
Compute the similarity matrix M s.t. $M[i,j]$ is the number of leaves where i and j are together normalized by the number of trees of the RF
for each $x \in \mathcal{U} = \mathcal{D} \setminus \mathcal{N}$ **do**
  for $i = 1$ to l **do**
    $$p(x \in N_i) = \frac{\frac{1}{|N_i|}\sum_{x_j \in N_i} M(x,x_j)}{\sum_{p=1}^{l} \frac{1}{|N_p|}\sum_{x_j \in N_p} M(x,x_j)}$$
  $$H(\mathcal{N}|x) = -\sum_{i=1}^{l} p(x \in N_i)\log_2 p(x \in N_i)$$
  $$E(x) = \sum_{i=1}^{l} i * p(x \in N_i)$$
return $\arg\max_{x \in \mathcal{U}} \frac{H(\mathcal{N}|x)}{E(x)}$

1 Background

2 Framework

3 Modification

4 Constraint Optimization

5 Active Constraint Selection

6 Experiments

7 conclusion

Methodology

- Evaluating single partition:
  - Functions measuring similarity between partitions:ARI,AMI,FMI...

$$ARI(\mathcal{P}_1, \mathcal{P}_2) = \cdots$$

- Evaluating all partitions:
  area under the budget curve(AUBC)
  two types:
  - $AUBC_{\text{quality}}$ when compare partitions with ground truth
  - $AUBC_{\text{similarity}}$ when compare partitions with previous one
- Compare different methods/parameters:
  Bayesian Pairwise Comparison(BPC)

Table 1 Dataset Characteristics, with N the number of instances, A the number of features and K the number of clusters or classes.

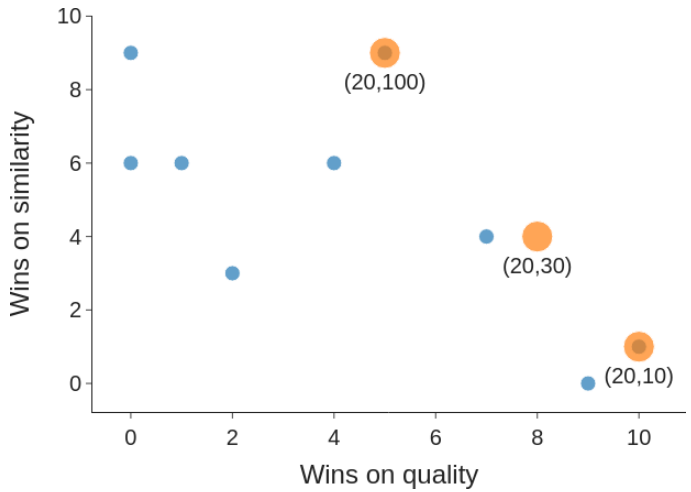
UCI				FCPS			
Name	(N, A, K)	Name	(N, A, K)	Name	(N, A, K)	Name	(N, A, K)
iris	(150, 4, 3)	ionosphere	(351, 34, 2)	lsun	(400, 2, 3)	chainlink	(1000, 3, 2)
wine	(178, 13, 3)	yeast	(1484, 8, 10)	target	(770, 2, 6)	wingnut	(1016, 2, 2)
sonar	(208, 60, 2)	statlog	(2310, 19, 7)	atom	(800, 3, 2)	engytime	(4096, 2, 2)
glass	(214, 9, 6)	Letters	(20000, 16, 26)				
ecoli	(336, 7, 8)	MNIST	(70000, 784, 10)				

Parameters Setting

- QUESTION:What effect do α, β have on clustering results?
- α : $\frac{\text{num}_{\text{anchors}}}{\text{size}_{\text{cluster}}}$
- β : $\frac{\text{num}_{\text{super-instace}}}{\text{size}_{\text{cluster}}}$
- dataset:all datasets

	$AUBC_{quality}$			$AUBC_{similarity}$		
(α, β)	ARI	AMI	FMI	ARI	AMI	FMI
$(0, 10)$	9 (1)	10 (1)	9 (1)	0	0	0
$(0, 30)$	2 (0)	1 (0)	1 (0)	3 (1)	2 (0)	1 (1)
$(0, 50)$	0	0	0	6 (3)	6 (2)	4 (2)
$(0, 100)$	0	4 (1)	3	9 (9)	9 (9)	9 (9)
$(5, 10)$	10 (3)	10 (3)	10 (3)	1 (0)	1 (0)	1 (0)
$(5, 30)$	7 (3)	5 (3)	7 (3)	4 (2)	4 (1)	3 (1)
$(5, 50)$	1 (0)	2 (0)	1 (0)	6 (4)	6 (3)	4 (2)
$(5, 100)$	5 (2)	4 (0)	5 (1)	9 (9)	9 (9)	9 (9)
$(20, 10)$	10 (7)	10 (6)	10 (6)	1 (1)	2 (1)	1 (0)
$(20, 30)$	8 (3)	6 (3)	7 (3)	4 (2)	4 (2)	3 (1)
$(20, 50)$	4 (1)	3 (1)	4 (1)	6 (5)	6 (4)	6 (4)
$(20, 100)$	5 (2)	7 (3)	5 (1)	9 (9)	9 (9)	9 (9)

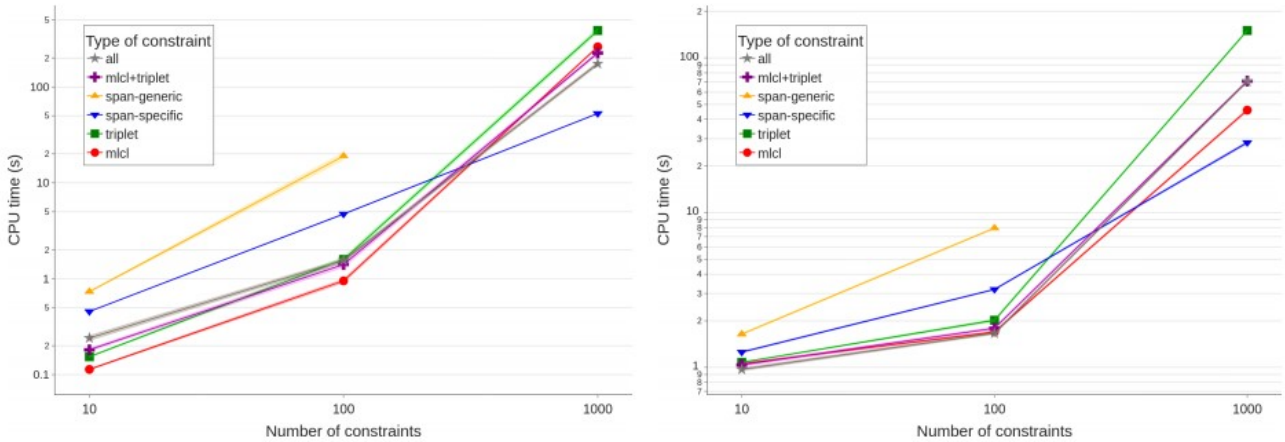
(a)



(b) use ARL

Runtime w.r.t Num&Types of Constraints

- randomly generate sets of four types of constraints
- initialize using KMeans
- average 90 runs
- Setting:$\alpha = 0\%, \beta = 100\%$



(a) `letters`. (b) `MNIST`.

Figure 6 Evolution of running time of our CP model for the two largest datasets when varying the number and type of constraints, with 95% confidence interval. CPU times are in log-scale.

Comparison with Other Soft Constraint Methods

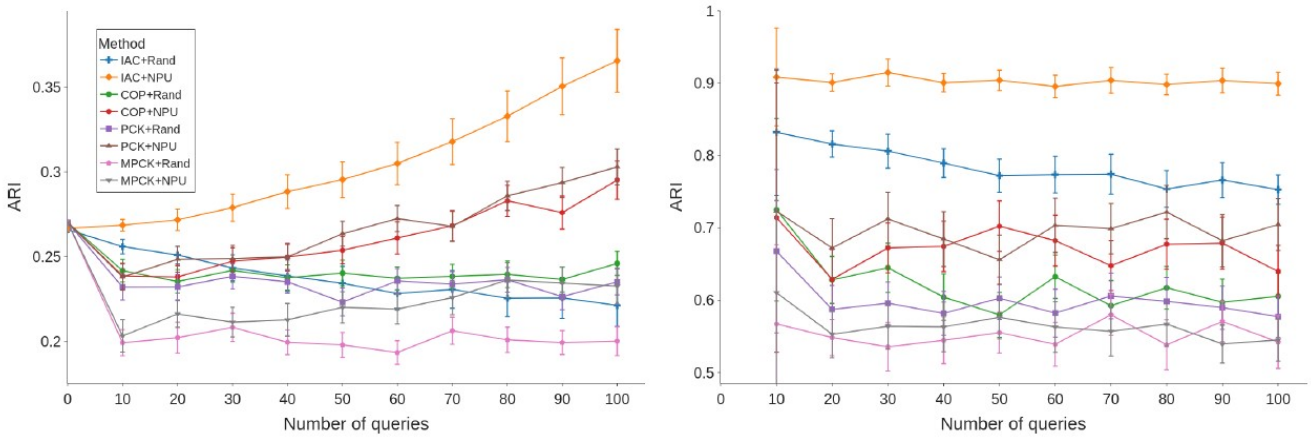
- Q:IAC 的 relaxing constraint 与其他方法的 soft constraints 相比如何？
- Setting:
  - IAC+Anchors:$\alpha = 20\%, \beta = 100\%$,IAC+Anchors+Super-Instances:$\alpha = 20\%, \beta = 30\%$
  - dataset:mk2

Table 2 Comparative study for clustering with pairwise constraints relaxation for the `mk2` dataset. Metrics are ARI with ground truth (Quality), ARI with unconstrained KMeans (Similarity), runtime and number of constraints relaxed in the solution (n_r).

	Test case with conflicts				Test case with $\delta = 94\%$			
	Quality	Similarity	Time	n_r	Quality	Similarity	Time	n_r
IAC+Anchors	0.576	**0.075**	5.262	49.7	0.309	**0.177**	3.051	60.1
IAC+Anchors+SI	**0.760**	0.024	4.868	49.83	0.393	0.108	2.972	60.1
PCK-Means	0.081	0.051	**3.639**	149.3	0.375	0.045	**2.834**	66.5
MPCK-Means	0.078	0.017	29.27	159.9	**0.406**	0.019	26.98	61.2

Comparison with Alternatives

- $\alpha = 20\%, \beta = 30\%$
- 10 iterations of selection-modification loop
- glass dataset



(a) Comparison to the ground truth.

(b) Comparison to the previous partition.

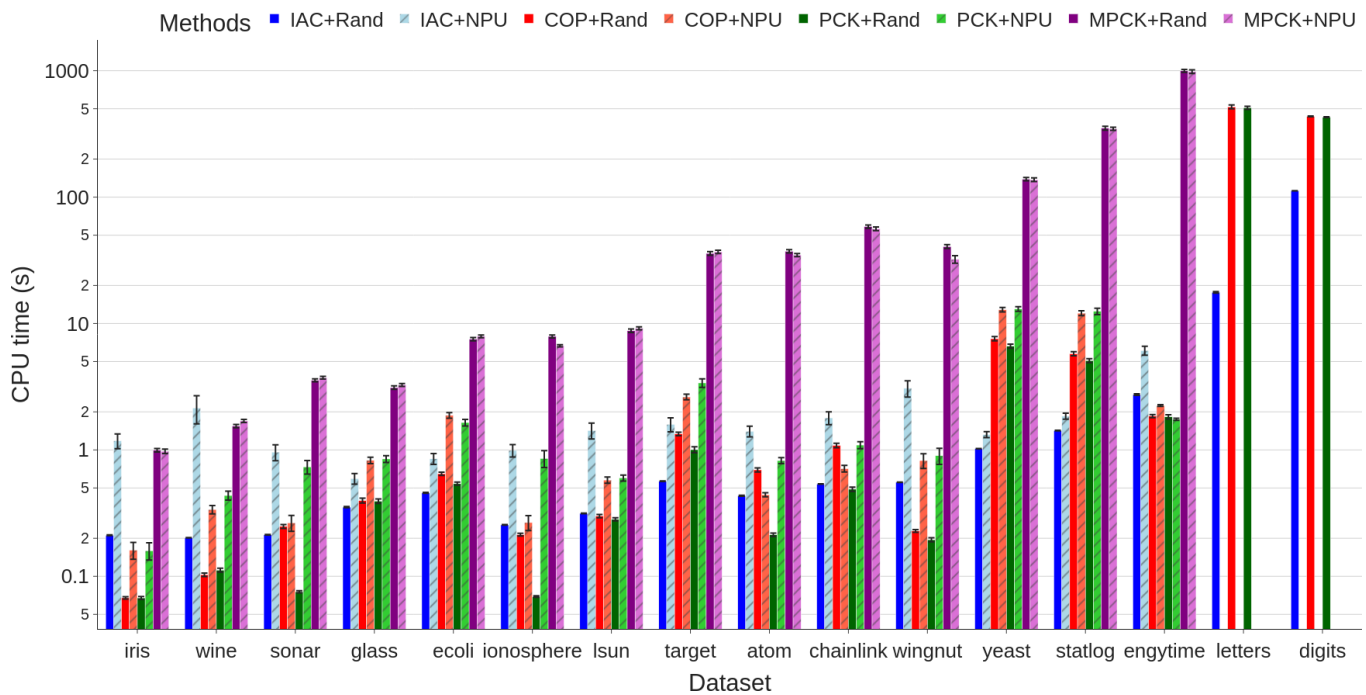
Figure 7 ARI scores of the partition at each iteration of selection-modification, compared to ground truth (left) or to the previous partition (right), for the `glass` dataset. All ARI scores are the mean and 95% confidence interval over 90 runs. Higher is better.

- more effective with NPU
- IAC+NPU has the highest ARI
- IAC+NPU has the best similarity

Comparison with Alternatives

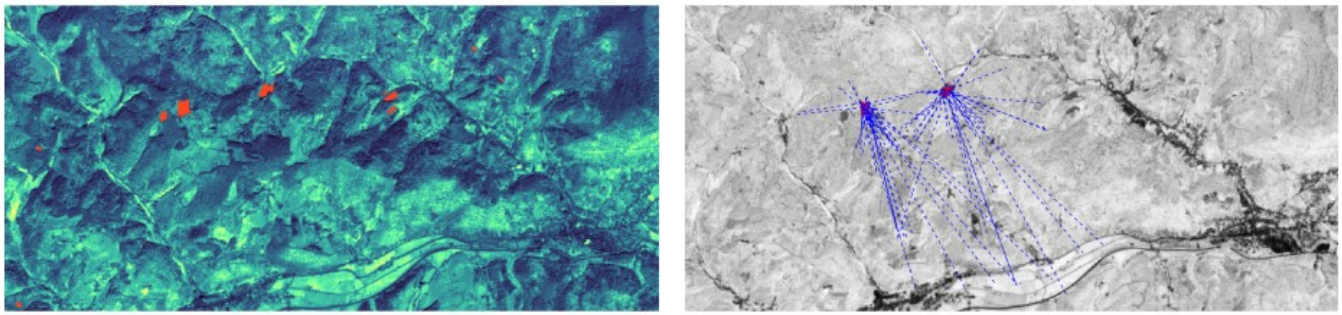
Modification time

- 虽然使用 NPU 能够使结果更好，但在较大数据 (e.g. letters) 上，耗时长，不适合用户交互。



Tree Cut Data

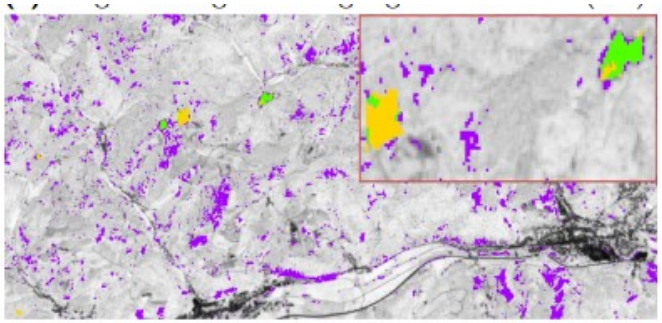
- Data:2016-2018 年间 11 张 724*337 的山脉卫星图，每个像素都与一个 NDVI(归一化植被指数) 值相关联。
- classes:vegetation, artifficial structure, tree cut zones
- 特点：tree cut zones 被领域专家精确标记，但是占比只有不到 0.3%(637 个点)，难以被无监督方法发现。
- Problem Definition：根据专家提供的 179 个 binary-constraints(图 b)，将 tree cut zones(图 a) 发现出来。



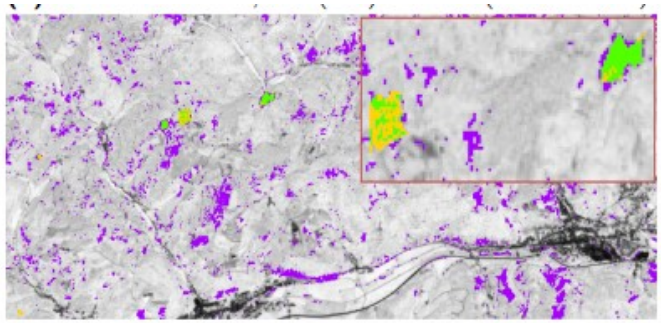
(a) Original image with highlighted tree cuts (red). **(b)** User constraints, ML (red) and CL (dashed blue).

Tree Cut Data

- 设置：$\beta = 1$, 迭代至所有输入约束被满足
- 使用 KMeans 初始化 15 个簇，面积最大的簇作为 positive cluster(图 c 着色部分)，其他簇作为 negative cluster(图 c 未着色部分)。将这种 binary partition 作为 IAC 输入。
- 结果为图 b：
  - green:true positive, yellor:false negetive, purple:false positive
  - 两处主要的 tree cut zone 结果：左边 TP 从 37/204→94/204, 右边 TP 增加了 10 个。



(c) Initial partition (with inset).



(d) Modified partition (with inset).

1 Background

2 Framework

3 Modification

4 Constraint Optimization

5 Active Constraint Selection

6 Experiments

7 conclusion

conclusion

- 提出了 Incremental and Active Clustering Framwork(IAC)，在 Incremental setting 中专家通过 manually 或 active method 来添加约束，从而使聚类有一定的连续性。
- IAC 的运行时间依赖 constraint selection 这一步，需要更多的研究来提出适合 incremental setting 的 active method。
- 如何重新利用 relaxed constraints 仍然是个 open question。

Thanks!