**Highlights**

- We propose a divide-and-conquer idea to geometric AL sampling.
- We provide the geometric insights for cooperating cluster boundary points in AL.
- An AL algorithm termed GAL is developed in this paper.
- We break the theoretical curse of uncertainty evaluation sampling by GAL algorithm.
- Experiments verify that GAL can be applied in multi-class settings of AL.

# A Divide-and-Conquer Approach to Geometric Sampling for Active Learning

Xiaofeng Cao*

*Advanced Analytics Institute, University of Technology Sydney*
*Email: xiaofeng.cao@student.uts.edu.au.*
*Address: 2 Blackfriars St, Chippendale NSW 2008*
*Phone:+61 0481126436.*

**Abstract**

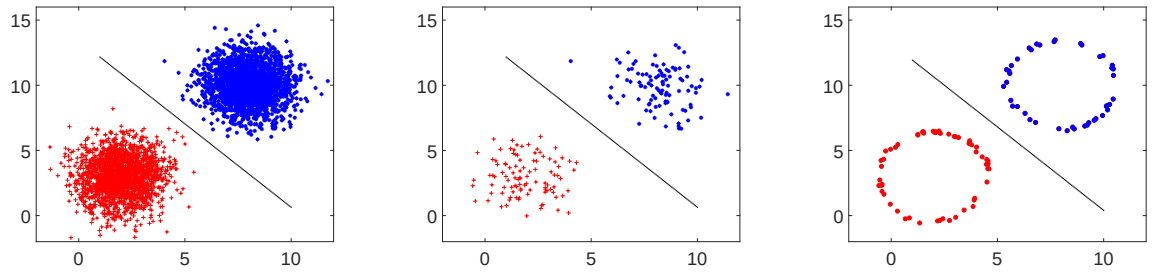
Active learning (AL) improves the current training model of the classifier, by querying the labels from the unlabeled data pool. The querying process is typically supervised by an uncertainty evaluation function. However, the uncertainty evaluation always suffers from performance degeneration when the initial labeled set has insufficient labels. To completely eliminate the dependence on the uncertainty evaluation sampling in AL, this paper proposes a divide-and-conquer idea that directly transfers the AL sampling as the geometric sampling over the clusters. By dividing the points of the clusters into cluster boundary and core points, we theoretically discuss their margin distance and hypothesis relationship. With the advantages of cluster boundary points in the above two properties, we propose a Geometric Active Learning (GAL) algorithm by knight's tour. Experimental studies of the two reported experimental tasks including cluster boundary detection and AL classification show that the proposed GAL method significantly outperforms the state-of-the-art baselines.

*Keywords:* Active learning, uncertainty evaluation, geometric sampling, cluster boundary.

## 1. Introduction

Active learning (Cohn et al., 1994) is explored to improve the prediction ability of the current classification model in supervised learning problems without sufficient labels. This study has been widely applied in various of learning scenarios where the unannotated data are abundant but annotating them is expensive and time-consuming, such as semi-supervised text classification (Hu et al., 2016), image annotation (Li et al., 2012), transfer learning (Guo et al., 2016), etc. Existing AL strategies focus on the construction of an uncertainty evaluation function which guides the subsequent sampling such as (Lewis & Gale, 1994), (Roy & McCallum, 2001), etc. However, this progress heavily depends on the label diversity and distribution features of the initial labeled

☆Fully documented templates are available in the elsarticle package on CTAN.




(a) Original data space (b) Cluster core points (c) Cluster boundary points

Figure 1: Motivation of our active learning work. In each sub-figure, the black line denotes the generated SVM classification model based on the data points in the figure. (a) Training the original data space. (b) Training the cluster core points. (c) Training the cluster boundary points. We observe that the generated classification lines of (c) are similar to the models of (a) and (b).

set. When the initial labeled set only has a few data, performance degeneration of the subsequent sampling would be inevitable.

Geometric sampling shows its power in various of domains such as fast SVM training (Tsang et al., 2005), Bayesian adversarial spheres algorithm (Bekasov & Murray, 2018), geometric deep learning (Fey et al., 2018), etc. Especially in large scale classification issue, Core Vector Machine (CVM) (Tsang et al., 2006) changed the SVM to a problem of minimum enclosing ball (MEB), which is popular in hard-margin support vector data description (SVDD) (Tax & Duin, 2004), and then iteratively calculated the ball center and radius in a $(1+\epsilon)$ approximation. In this process, the cluster boundary points located on the surface of each MEB are added into a special data collection called core sets. Trained by the detected core sets, the proposed CVM performed faster than the SVM and needed less support vectors. Especially in the Gaussian kernel, a fixed radius was used to simplify the MEB problem to the EB (Enclosing Ball), and accelerated the calculation process of the Ball Vector Machine (BVM) (Tsang et al., 2007). Without sophisticated heuristic searches in the kernel space, the training model, using points of high dimensional ball surface, can still be approximated to the optimal solution.

In this paper, we are motivated by the advantages of boundary points of CVM and propose a divide-and-conquer approach to geometric sampling for AL (see Figure 1). Underlying MEB model, we divide the data of each class into two types: cluster boundary and core points. In geometric description, cluster boundary points are located at the surface of one cluster and core points are distributed inside the cluster. To study the properties of the two types of points, we compare them from two-fold: margin distance (w.r.t. Lemma 1) and hypothesis relationship (w.r.t. Lemma 2). The conclusion shows that cluster boundary points play more important role in the construction of the classification hyperplane compared to core points in a geometrical perspective.

Our conquer step is to obtain the cluster boundary points. By setting a knight in the geometric space, the path disagreement of the tour helps us to differ from cluster boundary and core points. We assume the tour path is decided by the update process of traversing 1 to $k$ nearest neighbors ($k$NN) of the current tour position (data point). Their geometric disagreement in path length become the key of our detection method, i.e., the average tour path of boundary points are longer than that of the core points. With the above divide-and conquer analysis, we finally propose a Geometric Active Learning (GAL) algorithm by training the geometric cluster boundary points. The contributions of this paper are described as follows.

- We propose a divide-and-conquer idea to geometric AL sampling. It transfers the uncertain sampling space of AL into a set of the cluster boundary points.
- We provide the geometric insights for cooperating cluster boundary points in AL under the assumption of geometric classification.
- An AL algorithm termed GAL is developed in this paper. It samples independently without iteration and help from the labeled data.
- We break the theoretical curse of uncertainty evaluation sampling by GAL algorithm since it is neither a model-based nor label-based strategy with the fixed time and space complexities of $\mathcal{O}(NlogN)$ and $\mathcal{O}(N)$ respectively.
- A lot of experiments are conducted to verify that GAL can be applied in multi-class settings to overcome the binary classification limitation of many existing AL approaches.

The remainder of this paper is structured as follows. The related work is reported in Section 2. The preliminaries are described in Section 3 and the geometric insights on cluster boundary points in AL are presented in Section 4. The divide-and-conquer approach of knight's tour is presented in Section 5. The experiments and results are reported in Sections 6. The discussion is presented in Section 7. Finally, we conclude this paper in Section 8.

## 2. Related Work

In this section, we present the related work on active learning and cluster boundary research.

### *2.1. Active learning*

The learning goal of AL is to obtain a descried error rate by annotating as fewer queries as possible. To improve the performance of the current classification model, the AL learner (human expert) is allowed to pick

up a subset from an unlabeled data pool. Those data, which may largely affect the subsequent update of the learning model, are the primary goals of the learner. As a policy, accessing the unlabeled data pool to sample and querying their true labels with a given budge are approved. However, all the learners would face an awkward and difficult situation: how to fast select the descried data from the massive unlabeled data in the pool.

To resolve the above challenges, uncertainty evaluation (Lewis & Gale, 1994) was proposed to guide AL by selecting the most informative or representative instances in a given sampling scheme or distribution assumption, such as margin (Tong & Koller, 2001), uncertainty probability (Roy & McCallum, 2001), maximum entropy (Melville & Mooney, 2004), confused votes by committee (Seung et al., 1992), etc. For example, (Tong & Koller, 2001) proposes to select the data which is nearest to the current classification hyperplane, (Roy & McCallum, 2001) selects the data which can maximize the error rate change, (Melville & Mooney, 2004) selects the data with the maximum entropy of prediction probability, etc. Basically, these uncertainty-based AL algorithms aim to reduce the number of queries or converge the classifier quickly. Accompanied by multiple iterations, querying stops when the defined sampling number is met or a satisfactory model is found. It is thus these algorithms still need to traverse the whole data set repeatedly in this framework, although this technique performs well. However, they always suffer from one main limitation, that is, heuristically searching the whole data space to obtain the optimal sampling subset is impossible because of the unpredictable scale of the candidate set.

In practice, incorporating the unsupervised learning in the sampling process shows powerful advantages such as (Nguyen & Smeulders, 2004) (Kang et al., 2004) (Urner et al., 2013). It makes the learner solve the previous limitation be possible. One classical method (Dasgupta & Hsu, 2008) is performing the hierarchical clustering before sampling to improve th lower bound of the subsequent training performance. By setting up a probability condition, the learner is allowed to confidently annotate a number of subtrees with the label of the root note. When the clustering structure is perfect, it wold be positive for the sampling. However, an improper clustering results will mislead the annotation process. Then, performance degeneration of the subsequent sampling is inevitable.

### *2.2. Cluster boundary*

Cluster boundary points are a set of special objects distributed in the margin regions of each cluster. Their labels are given by the cluster structure and guide the clustering partition. However, those label assignations are uncertain. Nowadays, the practical advantage of the cluster boundary has been widely used in the latent virus carrier detection (Li et al., 2015), abnormal gene segment diagnosis (Qiu & Cao, 2016), etc.

With the prior experience in clustering algorithms, researchers firstly study the cluster boundary detection issue in the low dimensional space and propose a series of approaches, such as (Xia et al., 2006) (Qiu et al.,

2007) (Li et al., 2015) etc. In those proposed algorithms, BORDER firstly defines the cluster boundary points by measuring the density of their nearest neighbors, and uses the reverse $k$NN to obtain the complete boundary points, but with all the noises. To smooth the influence of noises, (Qiu et al., 2007) propose a detection algorithm termed BRIM via analyzing the balance property of the data distributed inside and outside the cluster. Because the extracted features are in low dimensional space, this algorithm could only be applied in two-dimension space. Moreover, the task of detecting the cluster boundary objects in high dimensional clusters is firstly studied in (Qiu & Cao, 2016) via utilizing the particle space inversion and Hopkins statistic. However, the devised Euclidean Gaussian filter function can not work well in very high-dimensional space because of the uncertainty of noises in the sparse distribution.

## 3. Preliminary

In this section, we first define the AL sampling by a family of linear functions. Then, we define the cluster boundary and core points by a group of density functions. Related definitions, main notations and variables are briefly summarized in Table I.

Table 1: A summary of notations

| Notation | Definition |
|---|---|
| $h_w, h_w^+, h_w^\beta, h_w^\zeta$ | classifiers |
| $error(h_w)$ | prediction error rate of $\mathcal{X}$ when training $h_w$ |
| $\mathcal{X}$ | data set |
| $N$ | data number of $\mathcal{X}$ |
| $N_l, N_u, N_q$ | number of labeled, unlabeled, queried data |
| $\mathcal{Y}$ | label set |
| $x_i, p$ | a data point in $\mathcal{X}$ |
| $\mathcal{X}_l$ | labeled data points in $\mathcal{X}$ |
| $\mathcal{X}_q$ | queried data points in $\mathcal{X}$ |
| $\mathcal{X}_t$ | training set after querying |
| $\mathcal{L}$ | distance function |
| $\zeta$ | core points |
| $\beta$ | cluster boundary points |
| $\eta$ | noises |
| $\chi$ | training set of $[\beta\ \zeta]$ |
| $\zeta^+$ | core points located inside the positive class |
| $\zeta^-$ | core points located inside the negative class |
| $\beta^*$ | cluster boundary points located near $h$ |
| $\eta$ | noises |
| $\zeta_1, \zeta_2$ | core points |
| $\beta_1, \beta_2$ | boundary points |
| $\eta_1, \eta_2, \eta_3$ | noises |
| $\rightarrow$ | approximation statement |
| $\leftarrow$ | assignment statement in algorithm |

Given $\mathcal{X}$ represents data space $\{x_1, x_2, x_3, ..., x_n\} \in \mathbb{R}^{n\times m}$, where $x_i = (x_{i1}, x_{i2}, x_{i3}, ..., x_{im})$ and the label space $\mathcal{Y} = (y_1, y_2, y_3, ..., y_n)$, considering the classification hypothesis:

$$h_w := w^T x + b, \tag{1}$$

where $w$ is the parameter vector and $b$ is the constant vector, here gives:

Definition 1. Active learning. Optimizing $w$ to get the minimum *RSS* (residual sum of squares)(Yu et al., 2006) (Zhang et al., 2011):

$$w^* = \underset{w}{\operatorname{argmin}}\{\sum_{i=1}^{n}(w^T x_i - y_i)^2\} \tag{2}$$

i.e.,

$$\begin{aligned} w^* =& (\mathcal{X}_t^T \mathcal{X}_t)^{-1} \mathcal{X}_t^T \mathcal{Y} \\ \text{s.t. } \mathcal{X}_t =& [\mathcal{X}_l \ \mathcal{X}_q], \end{aligned} \tag{3}$$

where $\mathcal{X}_l$ is the labeled data, $\mathcal{X}_q$ is the queried data, and $\mathcal{X}_t$ is the updated training set.

Definition 2. Cluster boundary point (Xia et al., 2006).

A boundary point $p$ is an object that satisfies the following conditions:

1. It is within a dense region $\mathbb{IR}$.

2. $\exists$ region $\mathbb{IR}'$ near $p$, $Density(\mathbb{IR}') \gg Density(\mathbb{IR})$ or $Density(\mathbb{IR}') \ll Density(\mathbb{IR})$.

Definition 3. Core point. A core point $p$ is an object that satisfies the following conditions:

1. It is within a dense region $\mathbb{IR}$.

2. $\exists$ an expanded region $\mathbb{IR}'$ based on $\mathbb{IR}$, $Density(\mathbb{IR}') - Density(\mathbb{IR}) \to 0$.

## 4. Geometric Insights

In clustering-based AL work, core points provide a little help for the parameter training of classifiers. Considering that cluster boundary points may provide decisive factors for the support vectors, CVM and BVM iteratively use the points distributed on the hyperplane of an enclosing ball to train fast core support vectors in large-scale data sets. Their significant success motivate the work of this paper.

To further show the importance of cluster boundary points, we (1) clarify the performance of training cluster boundary points in Section 4.1, (2) discuss the margin distance to the classification line or hyperplane of boundary and core points in Section 4.2, and (3) analyze the hypothesis relationship when training boundary and core points in Section 4.3, where the discussion cases of (2) and (3) are binary, and multi-class classifications of low and high-dimensional space.

### *4.1. Performance of cluster boundary*

In this section, we propose a geometrical perspective that the performance of the classification model is determined by the cluster boundary points. Our main theoretical result is summarized as follows.

**Proposition 1.** *Suppose that $\zeta$, $\beta$ respectively be a set of core points and cluster boundary points draw from a fixed geometrical cluster, $\Xi$ be their union of set that satisfies $\Xi$=[$\beta$ $\zeta$]. Let $h^{\Xi}$ be the classification hypothesis with respect to the training set $\Xi$, $h^{\beta}$ be another classification hypothesis with respect to the training set $\beta$. The following hods for the generalized error disagreement $\Delta^{'}$:*

$$\Delta^{'} = err(h^{\Xi}) - err(h^{\beta}) \rightarrow 0. \tag{4}$$

*where $\rightarrow$ denotes the approximation symbol.*

Our main theoretical results in Proposition 1 claim that the core points, distributed inside the center regions of any cluster, present little influences on training a descried hypothesis $h$. To demonstrate our insights, Lemma 1 and Lemma 2 provide theoretical supports in different geometrical views, where Lemma 1 proves that cluster boundary points have shorter margin distance to the geometric classification line or hyperplane compared with core points, and Lemma 2 proves the trained models generated from core points are a subset of the models generated from the boundary points. In next subsection, we respectively present the detailed proofs of the two lemmas in settings of binary, multi-class settings of low and high dimension space.

### *4.2. Margin distance*

Margin distance measures the distance to the classification line or hyperplane of one data point and we use $\mathcal{L}(.,.)$ to denote. The margin distance relations of boundary points and core points are described in the following lemma.

**Lemma 1.** *Suppose that $\zeta$, $\beta$ respectively be a set of core points and cluster boundary points draw from a fixed geometrical cluster. Let $\mathcal{L}(.,.)$ be the margin distance function. The margin distance of boundary points are shorter than the core points distributed in their local geometrical space, i.e.,*

$$\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h). \tag{5}$$

Lemma 1 is supported by Corollary 1 to 3 from different cases:

- Corollary 1: $\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in binary classification of low dimensional space, where Corollaries 1.1 and 1.2 prove Proposition 1 in adjacent classes and well-separated classes, respectively.
- Corollary 2: $\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in multi-class classification issue of low dimensional space.
- Corollary 3: $\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in high-dimensional space.

We now present detailed proofs for the above corollaries.

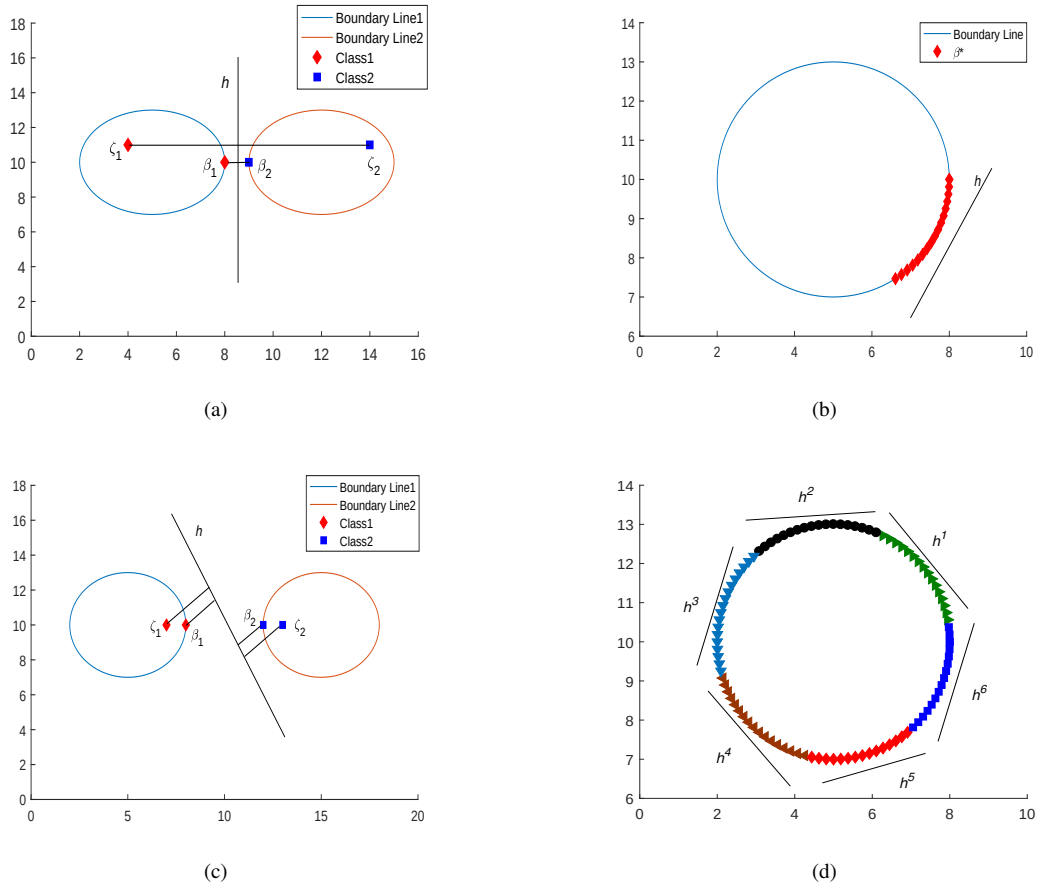



Figure 2: (a) An example of adjacent classes in two-dimensional space. $h$ denotes a linear classification hypothesis. The red diamonds denote samples of Class 1, the blue squares denote samples Class 2. $\zeta_1, \zeta_2$ are two core points and $\beta_1, \beta_2$ are two cluster boundary points. This figure illustrates Eq. (7) and the conclusion of it are $\mathcal{L}(\beta_1, h_w) < \mathcal{L}(\zeta_1, h_w)$ and $\mathcal{L}(\beta_2, h_w) < \mathcal{L}(\zeta_2, h_w)$. (b) An example of $\beta^*$ in the binary classification problem. This figure illustrates Eq. (11). (c)An example of well-separated classesin two-dimensional space. This figure illustrates Eq. (10). (d) An example of segmenting $\beta$ in the multi-class classification problem with $k = 6$.

**Corollary 1.** *$\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in binary classification of low dimensional space.*

Given two facts in the classification: (1) the data points far from $h$ usually have clear assigned labels with a high prediction class probability; (2) $h$ is always surrounded by noises and a part of the boundary points. Based on these facts, the proof is as follows.

**Corollary 1.1:** $\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in adjacent classes of low dimensional space.

*Proof.* Given any adjacent classes scenarios with binary labels ($\mathcal{Y} \in \{$-1,+1$\}$) such as Figure 2(a). Let $\zeta^+$ denote the core points located inside the positive class, $\zeta^-$ denote the core points located inside the negative class, $\beta^*$

denotes the cluster boundary points near $h$, and $\eta$ denote the noises near $h$. The RSS analysis in such classification scenarios satisfy:

$$\begin{cases} RSS(\zeta^+) = \sum_{i=1}^{N_{\zeta^+}} (w^T x - 1)^2 \to 0, \mathcal{X}_t = \zeta^+ \\ RSS(\zeta^-) = \sum_{i=1}^{N_{\zeta^-}} (w^T x + 1)^2 \to 0, \mathcal{X}_t = \zeta^- \\ RSS(\beta^*) = \sum_{i=1}^{N_{\beta^*}} (w^T x - 0)^2 \to 0, \mathcal{X}_t = \beta^* \\ RSS(\eta) = \sum_{i=1}^{N_\eta} (w^T x - 0)^2 \to 0, \mathcal{X}_t = \eta \end{cases} \tag{6}$$

where $N_{\zeta^+}, N_{\zeta^-}, N_{\beta^*}$, and $N_\eta$ denote their numbers of the four types of points. In most of classification issues, noises always have wrong guidance on model training. We therefore only focus on the differences between the core and boundary points, that is to say,

$$|h_w(\beta^*)|^2 - |h_w(\zeta)|^2 = (wx_{\beta^*}^T)^2 - (wx_\zeta^T)^2 = w^2(x_{\beta^*}^2 - x_\zeta^2) \to \epsilon_1 < 0. \tag{7}$$

where $\epsilon_1$ denotes a constant. In $\mathbb{R}$ space, the margin distance function between $x_i$ and $h$ could generalized as

$$\mathcal{L}(x_i, h_w) = \frac{|w_{i1}x_{i1} + w_{i2}x_{i2} + b|}{\sqrt{w_{i1}^2 + w_{i2}^2}}. \tag{8}$$

Considering that the classifier function is $h_w(x_i) = w_{i1}x_{i1} + w_{i2}x_{i2} + b$, we conclude $\mathcal{L}(\beta^*, h_w) < \mathcal{L}(\zeta, h_w)$. Then, Lemma 1 is as stated when $\beta = \beta^*$ (see Figure 2(b)). ■

**Corollary 1.2:** $\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in well-separated classes of low dimensional space.

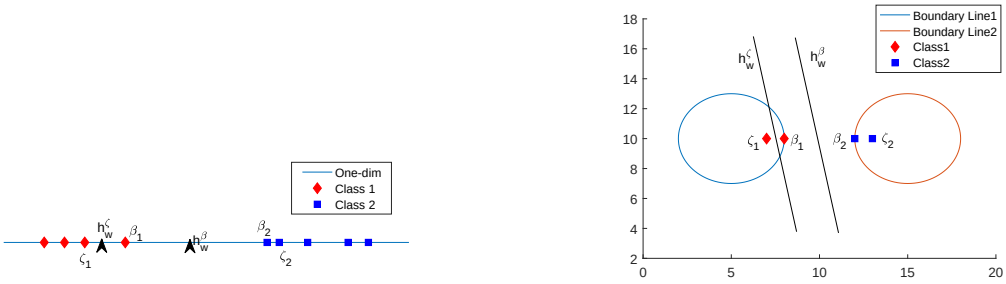
*Proof.* In the well-separated classes issue (see Figure 2(c)), the trained model based on any data points will lead to a strong classification result, that is to say, all AL approaches will perform well in this setting since:

$$\begin{cases} h_w(x_{\zeta^+}) - h_w(x_{\beta^+}) = wx_{\zeta^+}^T - wx_{\beta^+}^T \quad = w(x_{\zeta^+} - x_{\beta^+}) \to \epsilon_2 > 0. \\ h_w(x_{\zeta^-}) - h_w(x_{\beta^-}) = wx_{\zeta^-}^T - wx_{\beta^-}^T \quad = w(x_{\zeta^-} - x_{\beta^-}) \to \epsilon_3 < 0. \end{cases} \tag{9}$$

where $\beta^+$ denote a set of the cluster boundary points near $h$ in the positive class, $\beta^-$ denote a set of the cluster boundary points near $h$ in the negative class, $x_{\beta^+} \in \beta^+$, and $x_{\beta^-} \in \beta^-$. Let $\beta^* = \beta^+ \cup \beta^-$, $\zeta = \zeta^+ \cup \zeta^-$, the results of Eq. (8) and (9) still hold. ■

**Corollary 2.** $\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ *holds in multi-class classification in low dimensional space*

*Proof.* In this setting, $\mathcal{Y} \in \{0, 1, 2, ..., k\}$, the classifier set $H = \{h_w^1, h_w^2, h_w^3, ..., h_w^k\}$, and cluster boundary points are segmented into $k$ parts $\{\beta^1, \beta^2, \beta^3, ..., \beta^k\}$, where $\beta^i$ denotes the data points close to $h_w^i$, $i \in (1, k)$




(a) (b)

Figure 3: (a) An example of $h_w^\beta \subset h_w^\zeta$ in one-dimensional space. $h_w^\beta, h_w^\zeta$ are two point classifiers. (b) An example of $h_w^\beta \subset h_w^\zeta$ in two-dimensional space.

(see Figure 2(d))). Based on the result of Case 1, dividing the multi-class classification problem into $k$ binary classification problems, we can obtain:

$$|h_w(\beta^i)| < |h_w(\zeta^i)|, \forall i, \tag{10}$$

and

$$\mathcal{L}(\beta^i, h_w) < \mathcal{L}(\zeta^i), \forall i, \tag{11}$$

where $\zeta^i$ represents the core points near $h_w^i$. Then, the following holds:

$$\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h). \tag{12}$$

■

**Corollary 3.** *$\mathcal{L}(\beta, h) < \mathcal{L}(\zeta, h)$ holds in high-dimensional space.*

*Proof.* In a high-dimensional space, the distance function between $x_i$ and hyperplane $h_w$ could be extended as

$$\mathcal{L}(x_i, h_w) = |wx_i + C|(ww^T)^{-1/2}, \tag{13}$$

where $h_w(x_i) = wx_i + C$, and $C$ is a $m$-dimension vector. Because the above equation is the m-dimension extension of Eq. (9), the proof relating to low dimensional space is still valid in high-dimensional space. ■

### *4.3. Hypotheses relationship*

Lemma 2 describes this relationship of the hypotheses generated from the boundary and core points.

**Lemma 2.** *Suppose that $\zeta$, $\beta$ respectively be a set of core points and cluster boundary points draw from a fixed geometrical cluster. Let $h^{\zeta}$ be the hypothesis with respect to the training set $\zeta$, $h^{\beta}$ be another hypothesis with respect to the training set $\beta$. The following holds for*

$$h^{\beta} \subseteq h^{\zeta}. \tag{14}$$

It shows training models based on $\beta$ can predict $\zeta$ well, but the model based on $\zeta$ may sometimes not predict $\beta$ well. To prove this relation, we discuss it in three different cases:

- Corollary 4: $h^{\beta} \subseteq h^{\zeta}$ holds in binary classification of low dimensional space, where Corollary 4.1 and Corollary 4.2 prove Lemma 2 in one-dimension space and two-dimension space, respectively.
- Corollary 5: $h^{\beta} \subseteq h^{\zeta}$ holds in binary classification in high-dimensional space.
- Corollary 6: $h^{\beta} \subseteq h^{\zeta}$ holds in multi-class classification.

**Corollary 4.** *$h^{\beta} \subseteq h^{\zeta}$ holds in binary classification of low dimensional space.*

This corollary is supported by two different views in Corollary 4.1 and Corollary 4.2.

**Corollary 4.1:** $h^{\beta} \subseteq h^{\zeta}$ holds in linear one-dimension space.

*Proof.* Given point classifier $h_w^{\zeta}, h_w^{\beta}$ in the linear one-dimension space as described in Figure 3(a),

$$h_w^{\zeta} = \gamma, \gamma \in (\zeta_1, \zeta_2) \qquad or \qquad h_w^{\beta} = \gamma, \gamma \in (\beta_1, \beta_2) \tag{15}$$

where $\zeta_1, \zeta_2$ are core points. In comparison, the boundary points of $\beta_1, \beta_2$ have smaller distances to the optimal classification model $h_w^*$, i.e., $\zeta_1 < \beta_1, \zeta_2 < \beta_2$. Therefore, it is easy to conclude: $(\beta_1, \beta_2) \subseteq (\zeta_1, \zeta_2)$. Then, classifying $\zeta_1$ and $\zeta_2$ by $h_w^{\beta}$ is successful, but we cannot classify $\beta_1$ and $\beta_2$ by $h_w^{\zeta} = \gamma \in (\zeta_1, \beta_1)$, or $h_w^{\zeta} = \gamma \in (\beta_2, \zeta_2)$, respectively. ■

**Corollary 4.2:** $h^{\beta} \subseteq h^{\zeta}$ holds in two-dimensional space.

*Proof.* Given two core points $\zeta_1 = \{\zeta_{11}, \zeta_{12}\}, \zeta_2 = \{\zeta_{21}, \zeta_{22}\}$ in the two-dimensional space, the line segment $L_s^{\zeta}$ between them is described as follows:

$$\frac{y - \zeta_{12}}{\zeta_{22} - \zeta_{12}} = \frac{x - \zeta_{12}}{\zeta_{21} - \zeta_{11}}, x \in (\zeta_{11}, \zeta_{21}) \tag{16}$$

Training $\zeta_1$ and $\zeta_2$ obtain the following classification hypotheses:

$$\begin{aligned} & h_w^{\zeta}(x_i) = w_1^{\zeta} x_{i1} + w_2^{\zeta} x_{i2} + b, \{w_1^{\zeta}, w_2^{\zeta}, b\} \in (-\infty, +\infty) \\ & s.t. \quad h_w^{\zeta} \cap L_s^{\zeta}, tan\theta^{\zeta} = \left| \frac{\frac{\zeta_{12} - \zeta_{22}}{\zeta_{11} - \zeta_{21}} + \frac{w_1^{\zeta}}{w_2^{\zeta}}}{1 - \frac{\zeta_{12} - \zeta_{22}}{\zeta_{11} - \zeta_{21}} \frac{w_1^{\zeta}}{w_2^{\zeta}}} \right| \neq 0 \end{aligned} \tag{17}$$

where $\theta^{\zeta}$ is the angle between $h_w^{\zeta}$ (see Figure 3(b)).

Similarly, the classifier $h_w^{\beta}$ trained by $\beta_1 = \{\beta_{11}, \beta_{12}\}, \beta_2 = \{\beta_{21}, \beta_{22}\}$ is subject to:

$$h_w^{\beta} \cap L_s^{\beta}, \frac{y - \beta_{12}}{\beta_{22} - \beta_{12}} = \frac{x - \beta_{12}}{\beta_{21} - \beta_{11}}, x \in (\beta_{11}, \beta_{21}), \tag{18}$$

where $L_s^{\beta}$ is the line segment between $\beta_1$ and $\beta_2$. Intuitively, the difference of $h_w^{\beta}$ and $h_w^{\zeta}$ is their constraint equation. Because $(\beta_{11}, \beta_{21}) \subset (\zeta_{11}, \zeta_{21})$, we can conclude:

$$h_w^{\beta} \subset h_w^{\zeta}. \tag{19}$$

It aims to show $h_w^{\zeta}$ cannot classify $\beta_1$ and $\beta_2$ when $x \in (\zeta_{11}, \beta_{11})$ or $x \in (\beta_{11}, \zeta_{11})$ in the constraint equation. But for any $h_w^{\beta}$, it can classify $\zeta_1, \zeta_2$ correctly. ■

**Corollary 5.** *$h^{\beta} \subseteq h^{\zeta}$ holds in in high-dimensional space.*

*Proof.* Given two core points $\zeta_1 = \{\zeta_{11}, \zeta_{12}, \zeta_{13}, ..., \zeta_{1m}\}, \zeta_2 = \{\zeta_{21}, \zeta_{22}, \zeta_{23}, ..., \zeta_{2m}\}$, a bounded Hyperplane $S$ between them is:

$$S := \{x_i : x_{i1} \in (\zeta_{11}, \zeta_{21}), x_{i2} \in (\zeta_{12}, \zeta_{22}), ..., x_{im} \in (\zeta_{1m}, \zeta_{2m})\}. \tag{20}$$

Training the two data points can get the following classifier:

$$\begin{aligned} h_w^{\zeta}(x_i) &= \sum_{d=1}^{m} w_d^{\zeta} x_{id} + C, \{w_d^{\zeta}, C\} \in (-\infty, +\infty) \\ s.t. \quad & h_w^{\zeta} \cap S, cos\theta^{\zeta} = wv[(ww^T)^{1/2} + (vv^T)]^{-1/2} \end{aligned} \tag{21}$$

where $\theta^{\zeta}$ is the angle between $h_w^{\zeta}$ and $S$, $v$ is the normal vector of $S$. Given point $p$, which is located on $h_w^{\zeta}$, if $p_1 \in (\beta_{11}, \zeta_{11}), p_2 \in (\beta_{12}, \zeta_{22}), ..., p_m \in (\beta_{1m}, \zeta_{2m})$, in the positive class or $p_1 \in (\zeta_{11}, \beta_{11}), p_2 \in (\zeta_{12}, \beta_{22}), ..., p_m \in (\zeta_{1m}, \beta_{2m})$ in the negative class, $h_w^{\zeta}$ cannot predict $\beta_1$ and $\beta_2$ correctly. It can also be described as follows: if $h_w^{\zeta}$ segments the bounded hyperplane between $\zeta_1$ and $\beta_1$, or $\zeta_2$ and $\beta_2$, the trained $h_w^{\zeta}$ can not classify $\beta_1$ and $\beta_2$. Then Lemma 2 is as stated ■

**Corollary 6.** *$h^{\beta} \subseteq h^{\zeta}$ holds in multi-class classification issue.*

*Proof.* Follows the multi-class classification proof in Lemma 1, the multi-class problem could be segmented into $k$ parts of binary classification problems. ■

## 5. Geometric Active Learning by Knight's Tour

In our geometrical analysis, we divide the AL into a geometrical sampling process over a fixed cluster. The cluster boundary points, distributed in the margin regions of any class, have been demonstrated to provide more powerful support than core points, in terms of margin distance and hypothesis relationship. With this novel insight, in this section, we develop a conquer method to find this special set of points. However, the cluster boundary points always have multiple potential positions because of the uncertain locations of the classification hypotheses. As the diversity of the candidate positions of the cluster boundary points, recognizing all the potential positions can capture all the possible cluster boundary points against any multi-class scenarios.

Knight's tour is a classical path planning problem that requires the knight returns to the original starting point after traveling 64 chess lattices. Nowadays, this problem has became the path optimization in graph theory, and also been developed to a Markov chain problem in discrete state space. Setting the knight in data space $\mathcal{X}$ with $n$ samples, and its $k$-step transfer matrix $\mathcal{T}$ is:

$$\mathcal{T} = \begin{pmatrix} 0 & r_{t\times k}^{x_1\to x_2} & r_{t\times k}^{x_1\to x_3} & \cdots & r_{t\times k}^{x_1\to x_n} \\ r_{t\times k}^{x_2\to x_1} & 0 & r_{t\times k}^{x_2\to x_3} & \cdots & r_{t\times k}^{x_2\to x_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ r_{t\times k}^{x_n\to x_1} & r_{t\times k}^{x_n\to x_2} & r_{t\times k}^{x_n\to x_3} & \cdots & 0 \end{pmatrix} \tag{22}$$

where $r_{t\times k}^{x_i\to x_j}$ denotes that $x_i$ moves to $x_j$ in $k$ steps with a speed of $t$ steps once. When $t = 1$, $\mathcal{T}$ is the one-step transfer matrix of the knight's tour. Suppose that the knight begins the tour with a speed of $t = 1$ and a step length of $r_{1\times 1}^{x_i\to x_j} = ||x_i - x_j||_2$, where $||x_i - x_j||_2$ denotes the path length between $x_i$ and $x_j$. If the policy of the tour is to save the path cost, the knight needs to estimate each potential paths and takes a given probabilistic to select the subsequent position. Therefore, we propose the $1 \times 1$ transfer probabilistic matrix $\mathcal{P}$:

$$\mathcal{P} = \begin{pmatrix} 0 & p_{1\times 1}^{x_1\to x_2} & p_{1\times 1}^{x_1\to x_3} & \cdots & p_{1\times 1}^{x_1\to x_n} \\ p_{1\times 1}^{x_2\to x_1} & 0 & p_{1\times 1}^{x_2\to x_3} & \cdots & p_{1\times 1}^{x_2\to x_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{1\times 1}^{x_n\to x_1} & p_{1\times 1}^{x_n\to x_2} & p_{1\times 1}^{x_n\to x_3} & \cdots & 0 \end{pmatrix} \tag{23}$$

where $p_{1\times 1}^{x_i\to x_j}$ denotes the probability of moving into $x_j$ from $x_i$. We here define it by the ratio of the path length between $x_j$ from $x_i$ and all other possible paths, i.e., $p_{1\times 1}^{x_i\to x_j} = \frac{r_{1\times 1}^{x_i\to x_j}}{\sum_{v=1}^{n} r_{1\times 1}^{x_i\to x_v}}$, where $x_v \in \mathcal{X}$. Let $\mathcal{M}$ be the probabilistic transfer matrix produced by:

$$\mathcal{M} = \mathcal{T} \circ \mathcal{P}\mathcal{I}^{tr}, \tag{24}$$

where $\circ$ denotes the Hadamard product of two matrices, and $\mathcal{I} = [1, 1, 1, ..., 1]_{1\times n}$. Withe this operation, $\mathcal{M}$ denotes the length of the probabilistic transfer path when the current position of the tour is set from $x_i$,... , to $x_n$. Meanwhile, for any $x_i$, we have

$$\mathcal{M}_i = \sum_{j=1}^{n} \frac{||r_{1\times 1}^{x_i \to x_j}||_2^2}{\sum_{v=1}^{n} r_{1\times 1}^{x_i \to x_v}}. \tag{25}$$

$\mathcal{M}$ is a matrix with the size of $1 \times n$ and $\mathcal{M}_i$ is the probabilistic transfer path length of the tour when the knight is located at the position of $x_i$. This matrix characterizes the distribution features of the current location of the knight's tour. When the initial position of the tour is set in the central regions of the cluster, the knight would spend expensively to leave the cluster because the knight has multiple directions where can move into. However, if the knight is set in the boundary region of the cluster, the cost would decrease dramatically. Therefore, the tour path within a limited steps could intuitively reflect where the tour is, i.e., the boundary or the central regions of the cluster. With this policy, we further characterize the $k$ steps transfer path of each position by probability evaluation:

$$\mathcal{M}_i = \sum_{j=1}^{k} \frac{||r_{1\times 1}^{x_i \to M_i^j}|_2^2}{\sum_{v=1}^{k} r_{1\times 1}^{x_i \to M_i^v}} \tag{26}$$

where $M_i^j$ is the $j$nd neighbor of $x_i$ and we call $\mathcal{M}$ as the probabilistic tour matrix. The different between Eq. (25) and Eq. (26) is the tour space of the knight. In Eq. (25), $\mathcal{M}_i$ calculates the tour cost of leaving the cluster and the knight needs to visit $n$ positions. However, the tour cost in the local space characterizes the distribution features of cluster boundary and core points. Therefore, we limit the position numbers of the tour by a local variable $k$ in Eq. (26), which updates $x_j$ into $M_i^j$.

Based on the above definitions and analysis, we propose a Geometric Active Learning (GAL) algorithm. Its pseudo-code has been summarized in Algorithm 1. In its steps, Step 4 to Step 8 use the R-tree to calculate the $M$ matrix that denotes the $k$NN of each data in $\mathcal{X}$. The time complexity of this searching process approximates $nlog(n)$. Then, we calculate the probabilistic tour path of each data point using Eq. (26) and store these values in matrix $\mathcal{M}$. Step 9 sorts the values of matrix $\mathcal{M}$ by ascending. From a geometrical perspective, we divide the cluster into two regions: outer cluster collection $\mathcal{C}^{outer}$ and inner cluster collection $\mathcal{C}^{inner}$, where the outer cluster collection removes all noises from $\mathcal{X}$, the inner cluster collection covers all feasible core points from $\mathcal{X}$. Therefore, the cluster boundary collection of $\mathcal{X}$ includes the data belongs to $\mathcal{C}^{outer}$ but are not in $\mathcal{C}^{inner}$. To implement this process, we set two parameters named inner cluster ratio $\epsilon_1$ and outer cluster ratio $\epsilon_2$ to split $\mathcal{M}$. Let $\mathcal{M}'$ be a colon matrix via sorting matrix $\mathcal{M}$ by ascending, Step 8 to Step 14 describe this splitting process with the following policies: 1) for any data $x_i$, if its probabilistic transfer path length is shorter than $\mathcal{M}'_{\epsilon_1}$, it is a

**Algorithm 1:** Geometric Active Learning

**Input:** data set $\mathcal{X}$, number of queries $N_q$, nearest neighbor number $k$, inner cluster ratio $\epsilon_1 \in [0, 1]$ , outer cluster ratio $\epsilon_2 \in [0, 1]$, and $\epsilon_1 < \epsilon_2$.

**Initialize:** $\epsilon_1 \leftarrow \lceil n\epsilon_1 \rceil$, $\epsilon_2 \leftarrow \lceil n\epsilon_2 \rceil + N_q$, $\mathcal{M}' \leftarrow \emptyset$ .

Calculate the $k$NN matrix $M$ of $\mathcal{X}$ using R-tree search.

**for** *each data point* $x_i \in \mathcal{X}$ **do**

    Calculate $\mathcal{M}_i$ using Eq. (26).

**end**

Update $\mathcal{M}'$ via sorting $\mathcal{M}$ by ascending.

**while** $i \leq n$ **do**

    **if** $\mathcal{M}_i \leq \mathcal{M}'_{\epsilon_1}$ **then**

        Add $x_i$ into inner cluster collection $\mathcal{C}^{inner}$.

    **end**

    **if** $\mathcal{M}_i \leq \mathcal{M}'_{\epsilon_2}$ **then**

        Add $x_i$ into outer cluster collection $\mathcal{C}^{outer}$.

    **end**

    Return the collection of the boundary data by $\mathcal{C}^{outer} - \mathcal{C}^{inner}$.

**end**

data within the inner cluster, and 2) for any data $x_i$, if its probabilistic transfer path length is shorter than $\mathcal{M}'_{\epsilon_2}$, it is within the outer cluster. Finally, Step 15 returns the complement set of $\mathcal{C}^{outer}$ with respect to $\mathcal{C}^{inner}$.

## 6. Experiments

To demonstrate the effectiveness of our proposed GAL algorithm, we evaluate and compare the performance of the cluster boundary detection and AL classification with the existing algorithms in this section. The structure of this section is: Section 6.1 and 6.2 respectively describe the related baselines and tested data sets, Section 6.3 describes the preprocessing and evaluation, Section 6.4 describes the experimental settings, and Section 6.5 analyzes the results.

### *6.1. Baselines*

For the cluster boundary detection task, some baselines have been collected:

- **BORDER** (Xia et al., 2006) uses the reversal $k$NN approach to detect the cluster boundary based on a assumption of the reverse $k$NN number of cluster boundary points are less than that of core points. But its detection results always include all feasible noises because noises always have smaller number of reverse $k$NN , compared to other data.

- **BERGE**(Li et al., 2015) is the a iterative cluster boundary detection algorithm which uses evidence accumulation to start the detection, but the error rate always increases rapidly when labeling noises as cluster boundary points by mistake.

- **Spinver**(Qiu & Cao, 2016) algorithm, whose inspiration comes from spatial inversion of particle physics, is a high dimensional cluster boundary algorithm. It uses the Hopkins statistics to capture the neighborhood characteristics after smoothing noises by an Euclidean distance-based on Gaussian filtering function. But the Hopkins statistics prefers a balance class scenario.

For the classification task, several baselines also have been researched and will compare from GAL:

- **Random**, which uses a random sampling strategy to query unlabeled data, and can be applied to any AL task but with an uncertainty result.

- **Margin** (Tong & Koller, 2001), which selects the unlabeled data point with the shortest distance to the classification model, only can be supported by the SVM classification model.

- **Hierarchical** (Dasgupta & Hsu, 2008) sampling is a very different idea, compared to many existing AL approaches. It labels the subtree with the root node's label when the subtree meets the objective probability function. But incorrect labeling leads to a very bad classification result.

- **TED** (Yu et al., 2006) favors data points that are on the one side hard to-predict and on the other side representative for the rest of the experiments.

- **Re-active**(Lin et al., 2016) learning finds the data point which has the maximum influences on the future prediction result after annotating the selected data. This novel idea does not need to query the Oracle when relabeling, but needs a well-trained classification model at the beginning. Furthermore, its reported approach can't be applied in multi-class classification problems.

### *6.2. Data sets*

We synthesized and collected some emulated, benchmark data sets, respectively for the experiments described in this section, which are detailed as follows.

For the cluster boundary detection task, two clustering data sets named Aggregation and Flame, are used to show the concept of cluster boundary points. The other four classical clustering datasets Syn1- Syn4 are tested in the boundary detection experiment, where $n \times d$ denote the data set has $n$ samples with $d$ dimensions.

- **Syn1**:5400×2. The clusters are surrounded by a lot of noises.
- **Syn2**:4800×2.The circle cluster is embedded in the annulus cluster and a lot of noises connect them.
- **Syn3**:7832×2. There are two connected diamond clusters with multi-density.
- **Syn4**:5034×2. A lot of noises connect the different clusters.

The following datasets are real-world medical data sets.

- **Biomed** [1]:209×4. Medical data set. It has 134 normal objects and 75 virus infected objects. 30 virus carriers in the normal objects are defined as the cluster boundary of normal people.
- **Cancer** Qiu & Cao (2016):240×2. Medical data set. It has 241 malignant tumor objects and 75 benign tumor objects. 37 benign tumor objects which may become malignant tumor patients are cluster boundary objects of normal people.
- **Colon** [2]:240×2. Gene data set. 7 cluster boundary points.
- **Prostate**:240×2 Qiu & Cao (2016). Gene data set. 18 cluster boundary objects.

There are two image data sets in the target tracking field [3] and we sill use our GAL algorithm to capture the moving targets.

- **Waving Trees**:287×160. This comes from the data on the continuous monitoring of one building, including 7 captured images when a volunteer passes by the monitored area.
- **Moved Object**:1745×160. This comes from the data on the continuous monitoring of one office, including 363 captured images when a volunteer enters the office and leaves after staying some time.

There is also one sub-set of the Basel Face Model [4] in relation to the light test.

[1] http://lib.stat.cmu.edu/datasets/

[2] http://genomics-pubs.princeton.edu/oncology/affydata/

[3] http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm

[4] https://faces.dmi.unibas.ch/bfm/

- **Basel Face Model**:This is a popular 3D face model data set about multi-gestures and color change. The light sub-set has 4488 images, and all are stored with 500×500 pixels. We use the GAL algorithm to detect images with strong or dark light since only normal light images are useful in most real-world cases.

For the classification task of AL, we compare the bets classification results of different algorithms on some classical clustering data sets [5] and the letter recognition data set *letter*.

- **g2-2-30**:2048×2. There are 2 adjacent classes in the data set.
- **Flame**:240×2. It has 2 adjacent classes with similar densities.
- **Jain**:373×2. It has two adjacent classes with different densities.
- **Pathbased**:300×2. Two clusters are close and surround by a arc cluster.
- **Spiral**:312×2. There are three spiral curve cluster which are linear inseparable.
- **Aggregation**:788×2. There are 7 adjacent classes in the data set.
- **R15**:600×2. There are 7 separate clusters and 8 adjacent classes.
- **D31**:3100×2. It has 31 adjacent classes.
- **letter**:20000×16. It is a classical letter recognition data set with 26 English letters. We select 5 pairs letters which are difficult to distinguish from each other to test the above AL algorithms in a two-class setting. They are DvsP, EvsF, IvsJ, MvsN, UvsV, respectively. For multi-class test, we select A-D, A-H, A-L, A-P, A-T, A-X, A-Z, respectively. Of these, A-D is the letter set A to D, and A-H is the letter set A to H, ... , A-Z is the letter set A to Z. The seven multi-class sets have 4, 8, 12, 16, 20, 26 classes respectively.

In addition to the introduction for the tested data sets, all two-dimensional data sets are shown in Figure. 4.

### *6.3. Preprocessing and Evaluation*

The methods of preprocessing used in this paper are reported in this section. Application cases are: preprocessing methods (a) and (b) are used for the Colon and Prostate data sets, respectively since the compressed large domain will accelerate the calculation speed and reduce the memory consumption; pretreatment (c) is used to change the image type to number type and is used for Waving Trees, Moved Object and Basel Face Model. Here we detail the specific methods:

[5] http://cs.joensuu.fi/sipu/datasets/

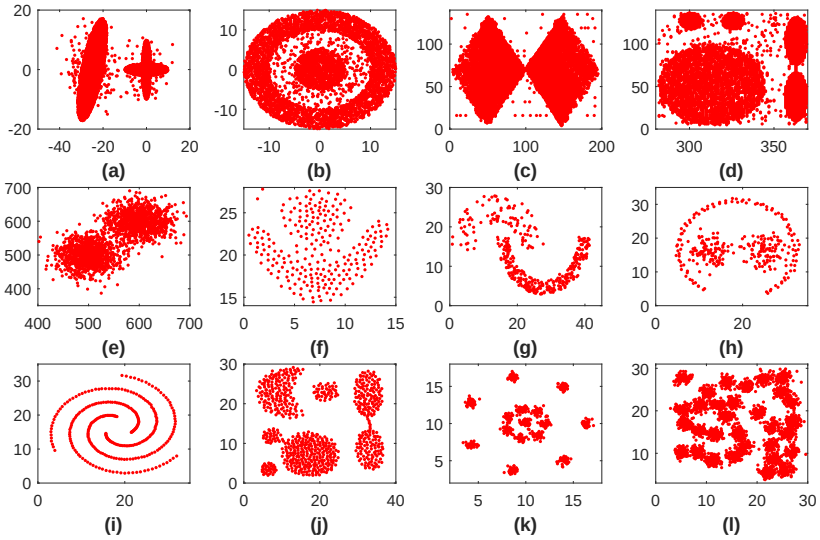



Figure 4: Classical clustering data sets. (a) Syn1 (b) Syn2 (c) Syn3 (d) Syn4 (e) g2-2-30 (f) Flame (g) Jain (h) Pathbased (i) Spiral (j)Aggregation (k) R15 (l) D31. The first four data sets are tested in the cluster boundary detection task and the others are tested in the classification task of AL.

(a) $x_{ij} = x_{ij}/10^3$, the value of each dimension of each data point is divided by $10^3$ ;

(b) $x_{ij} = x_{ij}/10^4$, the value of each dimension of each data point is divided by $10^4$;

(c) $G_j = \sum_{i=1}^{n} g_{ij}/n$, for each image, read the $n \times m$ grayscale matrix $g$ and compress it into a single-column matrix $G$ (i.e., with a size of $1 \times m$) with the average grayscale values.

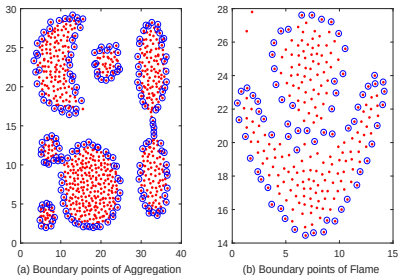



Figure 5: The marked cluster boundary point of Aggregation and Flame.

For the cluster boundary detection problem, we use the $F_1$ score to evaluate the detection result. This is a popular evaluation function in information retrieval which considers both precision $p$ and the recall $r$. Because the cluster boundary detection task is also a retrieval problem, we use it to evaluate our results. For the classification problem, we use accuracy to evaluate it.

### *6.4. Experimental setting*

We discuss the experimental setting of the compared algorithms over the synthetic and real data sets in this section.

- **Figure 5** marks the cluster boundary points on Syn1 and Syn2. It is used to show the definition of cluster boundary points.

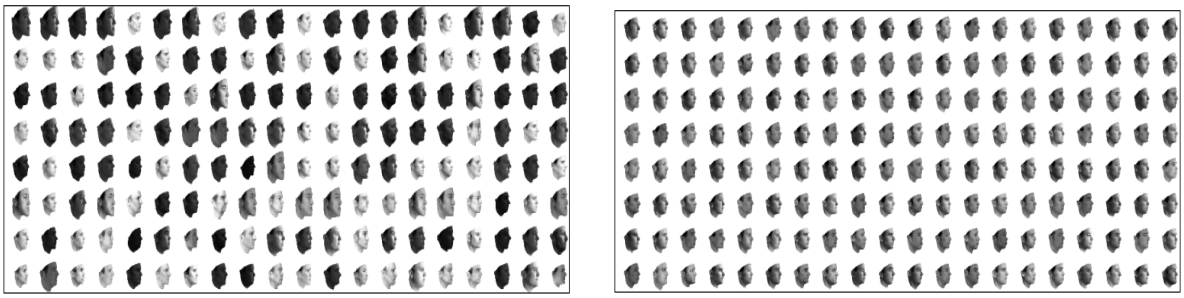
- **Table 2** reports the best cluster boundary detection result on different synthetic and real data sets. We have marked the highest $F_1$ scores of each group of experiment.

- **Figure 6(a)** shows the cluster boundary detection result on the light sub-data set of the Basel face. To compare the detected cluster boundary images, we also show the detection results for the core points using GAL in Figure 6(b).

- **Table 3** shows the classification results on some synthetic data sets. The specific experiment settings are as follows: (1) we use the MATLAB random function to implement the Random algorithm and calculate the mean and STD values after running it 100 times; (2) as the Margin, Hierarchical and Re-active algorithms all need the labeled data points to guide the training process, we select one data point from each class and query the Oracle, respectively. Similar, we test the algorithms 100 times and then calculate the mean and STD values in order to guarantee that the labeled set includes all the different label kinds of Oracle, or the algorithms will show poorer performance if we use random selection; (3) there are two important parameters for the TED algorithm: the kernel function parameter $\sigma$ and the regularization parameter for the kernel ridge regression $\lambda$. We use a super parameter $\sigma$=1.8 to generate the kernel matrix and train $\lambda$ from 0.01:0.01:1. The reason for this is that this parameter will provide important guidance for the sampling selection. After we test it many times, we limit its correct and stable range; (4) for our GAL algorithm, we train the parameters $k$ form 2:1:$\lfloor 5\%N \rfloor$ and boundary upper $\lambda_1$=$\lfloor 70\%N \rfloor$:1:$N$ to record the classification result. Because $\lambda_1$ segments the core points and boundary points, we use a super parameter $\lambda_1$=$\lfloor 70\%N \rfloor$ to begin the training. The conclusion that there are at least 70% N data points as core points in the data set comes from our published papers (Qiu & Cao, 2016) and experience summary. The classifier trained in the classification experiment is LIBSVM (Chang & Lin, 2011).

### *6.5. Results*

In Figure 5, we use the GAL algorithm to detect the cluster boundary points and mark them by the blue circles. Observing the marked data points, cluster boundary points not only can segment the different cluster structure but also can help to get the complete cluster/class structure after filling up the core points into the boundary internal area. An observation of the experimental results in Table 2 shows: (1) although the precision of BORDER is high, the recall rate also is high since it cannot smooth the noises and then the $F_1$ scores are also low in the synthetic data sets: Syn1 to Syn4. But this situation is reversed in the real-world data sets with little



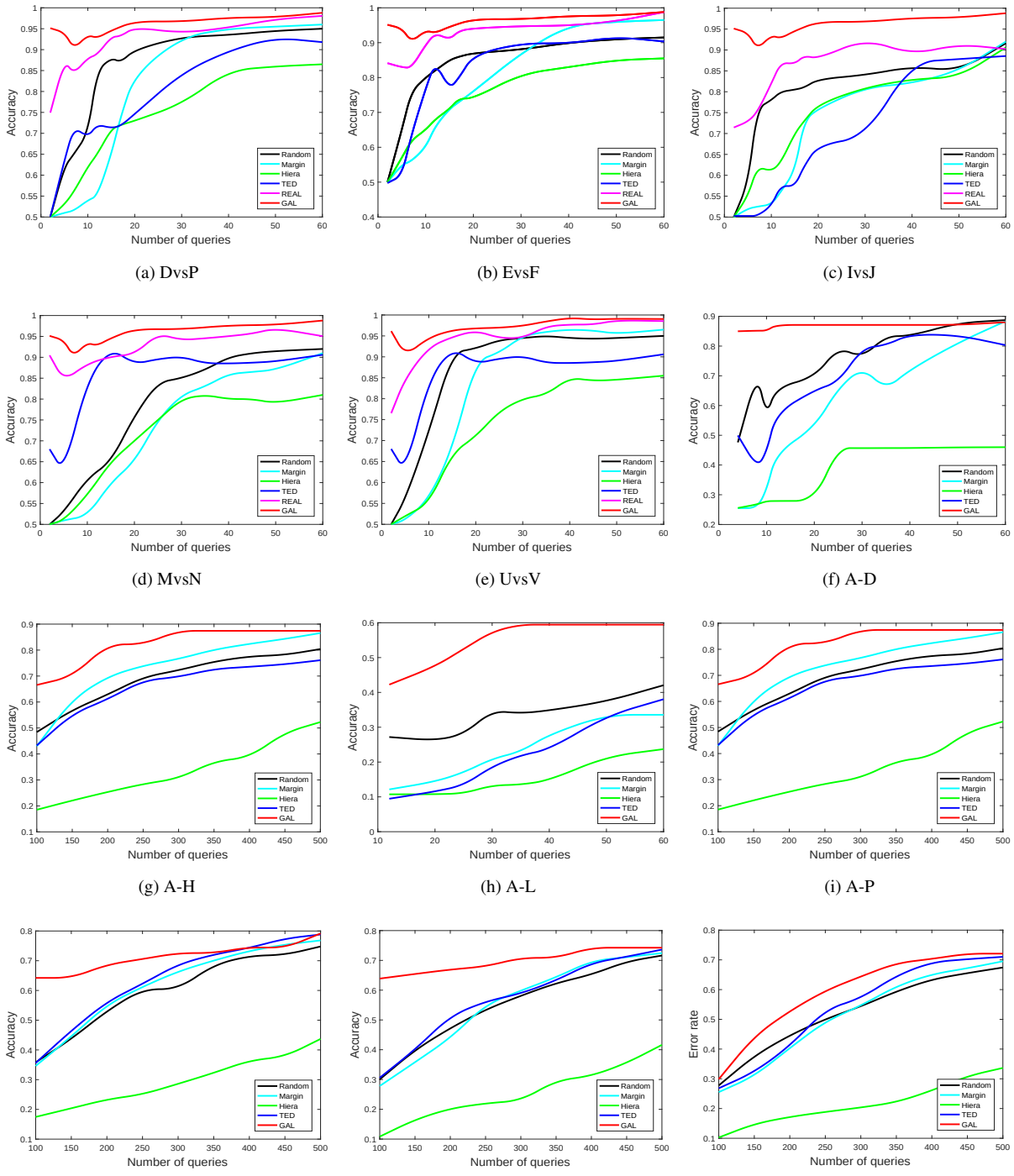
(a) Cluster boundary images

(b) Cluster core images

Figure 6: The cluster boundary and core images detection results using GAL algorithm on the light subset of Basel face model.

or no noise, such as the Colon, Prostate, and Waving Trees data sets. (2) The BERGE algorithm annotates some cluster boundary points to guide the following iterative detection where the error rate may rise rapidly when the annotation action is wrong. So, the noises increase the risk level and it is also sensitive to noises. (3) The Spinver algorithm uses a Gaussian filtering function to smooth noises and get a better detection result, compared to the above two approaches. (4) For GAL, we use the idea of object separation to detect the cluster boundary points, which is not sensitive to noises and dimensions since sorting is its main idea. The detection results also show our proposed algorithm outperforms Spinver. In Figure 6, the detected images of Figure 6(a) are the faces with normal light and the detected images of Figure 6(b) are the faces with strong or weak light. This is an interesting application for face recognition problems which will help to detect abnormal images in the resident information database, illegal document photos, etc.

Table 3 reports the classification results of different AL approaches in the two-dimension data sets. We mark some specific results to analyze the algorithm characteristics. The observation shows: (a) Random provides a fast sampling strategy which is not sensitive to data number and dimensions or class number. But its performance is always bad for the first query as it cannot select valuable data points using a random strategy. (b) Margin is a popular AL approach that selects the data points which are closest to the current classification plane. The results in the published papers show it is a good AL approach. However, our paper is the first to use the challenging two-dimension clustering data sets in AL and the experiment results show a drawback of Margin. That is, it has well-separated class bias, as it always selects the data points between adjacent classes since the calculated distance is small. Therefore, an unfair and unreasonable sampling strategy always selects the data points distributed in the most adjacent area in Jain, then returns a bad classification result (refer to the boxed results for Margin in the Jain data set in Table 3); (c) Hierarchical is a special AL approach which uses pre-clustering to judge whether the subtree nodes could be labeled with the label of the root node. In the collected test results of Table 3, it could




(a) DvsP (b) EvsF (c) IvsJ

(d) MvsN (e) UvsV (f) A-D

(g) A-H (h) A-L (i) A-P

(j) A-T (k) A-X (l) A-Z

Figure 7: The SVM classification results of different AL approaches on the letter data set. (a)-(e) are the binary classification settings. (f)-(l) are the multi-class settings. The class number respectively are 4, 8, 12, 16, 20, 24, and 26. In all sub figures, Hiera is the abbreviation of Hierarchical, REAL is the abbreviation of Re-active.

Table 2: The best cluster boundary detection results of the four algorithms on the synthetic and real data sets.

| Datasets | Dimension | Algorithm | Real.boun | Num.det | Num.C | Precision | Recall | $F_1$ |
|---|---|---|---|---|---|---|---|---|
| Syn1 | 2 | BORDER | 1077 | 1252 | 831 | 0.6637 | 0.7716 | 0.7136 |
| | | BERGE | | 1250 | 940 | 0.7520 | 0.8728 | 0.8079 |
| | | Spinver | | 1049 | 993 | 0.9466 | 0.9220 | 0.9341 |
| | | GAL | | 1043 | 996 | 0.9549 | 0.9248 | **0.9396** |
| Syn2 | 2 | BORDER | 1204 | 1802 | 1089 | 0.6043 | 0.9045 | 0.7246 |
| | | BERGE | | 1456 | 1098 | 0.7541 | 0.9120 | 0.8256 |
| | | Spinver | | 1264 | 1111 | 0.8790 | 0.9228 | 0.9003 |
| | | GAL | | 1163 | 1040 | 0.8942 | 0.9302 | **0.9118** |
| Syn3 | 2 | BORDER | 640 | 723 | 540 | 0.7469 | 0.8438 | 0.7924 |
| | | BERGE | | 662 | 532 | 0.8036 | 0.8313 | 0.8172 |
| | | Spinver | | 611 | 542 | 0.8871 | 0.8469 | 0.8665 |
| | | GAL | | 632 | 580 | 0.9177 | 0.9063 | **0.9120** |
| Syn4 | 2 | BORDER | 538 | 669 | 445 | 0.6366 | 0.8271 | 0.7195 |
| | | BERGE | | 553 | 472 | 0.8535 | 0.8773 | 0.8652 |
| | | Spinver | | 540 | 482 | 0.8926 | 0.8959 | 0.8942 |
| | | GAL | | 540 | 496 | 0.9185 | 0.9219 | **0.9202** |
| Biomed | 4 | BORDER | 30 | 26 | 23 | 0.8846 | 0.7667 | 0.8214 |
| | | BERGE | | 27 | 24 | 0.8889 | 0.8000 | 0.8421 |
| | | Spinver | | 29 | 27 | 0.9310 | 0.9000 | 0.9153 |
| | | GAL | | 29 | 28 | 0.9655 | 0.9333 | **0.9491** |
| Cancer | 10 | BORDER | 37 | 37 | 28 | 0.7568 | 0.7568 | 0.7568 |
| | | BERGE | | 37 | 30 | 0.8108 | 0.8108 | 0.8108 |
| | | Spinver | | 35 | 34 | 0.9714 | 0.9789 | 0.9444 |
| | | GAL | | 36 | 35 | 0.9722 | 0.9459 | **0.9589** |
| Colon | 2000 | BORDER | | 7 | 7 | 1.0000 | 1.0000 | **1.0000** |
| | | BERGE | | 6 | 5 | 0.8333 | 0.7143 | 0.7692 |
| | | Spinver | | 7 | 7 | 1.0000 | 1.0000 | **1.0000** |
| | | GAL | | 7 | 7 | 1.0000 | 1.0000 | **1.0000** |
| Prostate | 10,509 | BORDE | | 19 | 18 | 0.9474 | 1.0000 | **0.9730** |
| | | BERGE | | 17 | 16 | 0.9412 | 0.8889 | 0.9143 |
| | | Spinver | | 18 | 18 | 1.0000 | 1.0000 | **1.0000** |
| | | GAL | | 18 | 18 | 1.0000 | 1.0000 | **1.0000** |
| Waving Trees | 160 | BORDE | | 17 | 17 | 1.0000 | 1.0000 | **1.0000** |
| | | BERGE | | 17 | 15 | 0.8824 | 0.8824 | 0.8824 |
| | | Spinver | | 17 | 17 | 1.0000 | 1.0000 | **1.0000** |
| | | GAL | | 17 | 17 | 1.0000 | 1.0000 | **1.0000** |
| Moved Object | 160 | BORDE | | 363 | 222 | 0.6116 | 0.6116 | 0.6116 |
| | | BERGE | | 363 | 250 | 0.6887 | 0.6887 | 0.6887 |
| | | Spinver | | 363 | 222 | 0.6116 | 0.6116 | 0.6116 |
| | | GAL | | 363 | 352 | 0.9697 | 0.9697 | **0.9697** |

obtain good classification results when the data sets are well-structured classes. For example, it outperforms the other algorithms when labeling 1% data points in the data set R15; (d) Selecting the most uncertain data points to label also is applied in the TED approach, which also pays attention to representative data points. But in our

Table 3: The statistical results (mean±std) of different AL algorithms on classical cluster data sets.

| Data sets | Num_C | Algorithms | Number of queries (percentage of the data set) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1% | 5% | 10% | 15% | 20% | 30% | 40% | 50% | 60% |
| Biomed | 2 | Random | .516±.026 | .546±.012 | .603±.028 | .652±.029 | .693±.031 | .767±.026 | .815±.026 | .849±.021 | .881±.022 |
| | | Margin | .500±.000 | .509±.015 | .551±.047 | .590±.076 | .644±.103 | .709±.153 | .822±.139 | .882±.161 | .927±.188 |
| | | Hierarchical | .504±.000 | .550±.000 | .585±.000 | .615±.000 | .668±.000 | .774±.014 | .847±.000 | .920±.011 | .974±.000 |
| | | TED | .610±.000 | .619±.009 | .651±.003 | .759±.006 | .848±.007 | .875±.005 | .901±.005 | .964±.005 | .972±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .724±.163 | .725±.022 | .790±.021 | .825±.018 | .886±.012 | .909±.013 | .927±.011 | .994±.008 | 1.00±.000 |
| Cancer | 2 | Random | .516±.026 | .546±.012 | .603±.028 | .652±.029 | .693±.031 | .767±.026 | .815±.026 | .849±.021 | .881±.022 |
| | | Margin | .500±.000 | .509±.015 | .551±.047 | .590±.076 | .644±.103 | .709±.153 | .822±.139 | .882±.161 | .927±.188 |
| | | Hierarchical | .504±.000 | .550±.000 | .585±.000 | .615±.000 | .668±.000 | .774±.014 | .847±.000 | .920±.011 | .974±.000 |
| | | TED | .610±.000 | .619±.009 | .651±.003 | .759±.006 | .848±.007 | .875±.005 | .901±.005 | .964±.005 | .972±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .724±.163 | .725±.022 | .790±.021 | .825±.018 | .886±.012 | .909±.013 | .927±.011 | .994±.008 | 1.00±.000 |
| g2-2-30 | 2 | Random | .516±.026 | .546±.012 | .603±.028 | .652±.029 | .693±.031 | .767±.026 | .815±.026 | .849±.021 | .881±.022 |
| | | Margin | .500±.000 | .509±.015 | .551±.047 | .590±.076 | .644±.103 | .709±.153 | .822±.139 | .882±.161 | .927±.188 |
| | | Hierarchical | .504±.000 | .550±.000 | .585±.000 | .615±.000 | .668±.000 | .774±.014 | .847±.000 | .920±.011 | .974±.000 |
| | | TED | .610±.000 | .619±.009 | .651±.003 | .759±.006 | .848±.007 | .875±.005 | .901±.005 | .964±.005 | .972±.000 |
| | | Re-active | .506±.008 | .531±.029 | .554±.052 | .593±.065 | .634±.058 | .744±.060 | .715±.047 | .811±.000 | .816±.000 |
| | | GAL | .724±.163 | .725±.022 | .790±.021 | .825±.018 | .886±.012 | .909±.013 | .927±.011 | .994±.008 | 1.00±.000 |
| Flame | 2 | Random | .670±.142 | .794±.106 | .904±.059 | .944±.036 | .958±.025 | .976±.014 | .984±.008 | .987±.005 | .990±.006 |
| | | Margin | .499±.137 | .596±.102 | .740±.162 | .872±.158 | .930±.159 | .935±.145 | .961±.120 | .963±.109 | .944±.165 |
| | | Hierarchical | .720±.041 | .607±.042 | .855±.062 | .972±.010 | .999±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| | | TED | .829±.000 | .950±.006 | .974±.006 | .988±.006 | .991±.000 | .995±.001 | .996±.002 | .996±.002 | .998±.000 |
| | | Re-active | .553±.154 | .804±.120 | .917±.090 | .966±.045 | .974±.045 | .993±.006 | .993±.027 | .996±.004 | .997±.004 |
| | | GAL | .887±.004 | .976±.008 | .983±.005 | .988±.004 | .991±.002 | .995±.002 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| Jain | 2 | Random | .659±.180 | .773±.042 | .816±.041 | .848±.041 | .881±.040 | .928±.028 | .958±.024 | .974±.015 | .981±.015 |
| | | Margin | .258±.003 | .270±.074 | .382±.211 | .545±.306 | .572±.310 | .627±.347 | .623±.340 | .721±.347 | .736±.352 |
| | | Hierarchical | .325±.013 | .295±.008 | .297±.010 | .636±.022 | .873±.024 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| | | TED | .739±.000 | .764±.006 | .837±.018 | .932±.019 | .978±.018 | .998±.002 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| | | Re-active | .666±.163 | .748±.036 | .791±.027 | .836±.041 | .899±.045 | .994±.022 | .998±.008 | 1.00±.000 | 1.00±.000 |
| | | GAL | .768±.007 | .915±.026 | .963±.018 | .977±.013 | .989±.009 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| Pathbased | 3 | Random | .447±.157 | .533±.089 | .719±.096 | .833±.063 | .891±.046 | .940±.046 | .958±.016 | .969±.014 | .976±.010 |
| | | Margin | .366±.000 | .368±.016 | .407±.087 | .481±.151 | .686±.230 | .875±.209 | .960±.151 | .962±.148 | .988±.081 |
| | | Hierarchical | .488±.027 | .500±.017 | .547±.024 | .717±.028 | .749±.023 | .861±.022 | .949±.015 | .970±.013 | 1.00±.000 |
| | | TED | .356±.000 | .582±.023 | .875±.032 | .933±.008 | .941±.005 | .987±.009 | .997±.002 | 1.00±.000 | 1.00±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .748±.004 | .811±.048 | .920±.038 | .950±.019 | .959±.012 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| Spiral | 3 | Random | .352±.023 | .493±.049 | .634±.061 | .757±.059 | .830±.051 | .918±.034 | .955±.024 | .977±.017 | .988±.011 |
| | | Margin | .337±.005 | .344±.015 | .408±.062 | .513±.101 | .630±.144 | .893±.180 | .964±.119 | .965±.126 | .990±.034 |
| | | Hierarchical | .380±.024 | .486±.044 | .498±.046 | .525±.062 | .627±.044 | .653±.048 | .770±.055 | .774±.062 | .865±.039 |
| | | TED | .355±.000 | .678±.011 | .751±.039 | .828±.039 | .896±.003 | .920±.002 | .960±.000 | .990±.003 | .998±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .427±.017 | .685±.090 | .830±.097 | .872±.082 | .919±.063 | .963±.038 | .990±.021 | .998±.006 | 1.00±.000 |
| Aggregation | 7 | Random | .339±.101 | .583±.062 | .775±.047 | .868±.031 | .923±.023 | .972±.013 | .987±.006 | .993±.003 | .996±.000 |
| | | Margin | .215±.000 | .355±.092 | .707±.153 | .964±.098 | .995±.044 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| | | Hierarchical | .471±.038 | .578±.016 | .651±.009 | .695±.010 | .961±.009 | .987±.005 | .990±.005 | .992±.003 | .997±.000 |
| | | TED | .379±.002 | .646±.019 | .948±.009 | .968±.001 | .999±.001 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .808±.081 | .926±.016 | .964±.017 | .970±.022 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 | 1.00±.000 |
| R15 | 15 | Random | .337±.053 | .826±.067 | .955±.045 | .986±.015 | .992±.000 | .993±.000 | .993±.000 | .994±.000 | .994±.000 |
| | | Margin | .073±.020 | .393±.057 | .989±.003 | .997±.000 | .998±.000 | .998±.000 | .998±.000 | .998±.000 | .998±.000 |
| | | Hierarchical | .929±.010 | .990±.000 | .991±.000 | .995±.000 | .995±.000 | .996±.000 | .996±.000 | .996±.000 | .996±.000 |
| | | TED | .397±.002 | .984±.004 | .991±.002 | .994±.001 | .998±.000 | .998±.000 | .998±.000 | .998±.000 | .998±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .400±.000 | .989±.007 | .997±.001 | .997±.000 | .998±.000 | .998±.000 | .998±.000 | .998±.000 | .998±.000 |
| D31 | 31 | Random | .401±.040 | .899±.027 | .955±.005 | .964±.003 | .968±.000 | .971±.000 | .973±.000 | .974±.000 | .975±.000 |
| | | Margin | .067±.015 | .556±.064 | .968±.003 | .980±.000 | .983±.000 | .985±.000 | .986±.000 | .987±.000 | .988±.000 |
| | | Hierarchical | .879±.009 | .911±.006 | .951±.003 | .965±.000 | .976±.000 | .980±.000 | .981±.000 | .982±.000 | .981±.000 |
| | | TED | .936±.000 | .944±.001 | .960±.000 | .972±.000 | .980±.000 | .982±.000 | .979±.000 | .980±.000 | .980±.000 |
| | | Re-active | - | - | - | - | - | - | - | - | - |
| | | GAL | .954±.000 | .969±.000 | .974±.000 | .981±.000 | .982±.000 | .989±.000 | .989±.000 | .989±.000 | .989±.000 |

experiment, it is very slow and sensitive to parameters ( see its std values in each result). (e) Re-active selects the data points which have the greatest error disagreement on the labeled data after assigning the queried data with different labels. However, noises may be their main sampling objects whatever label they will be assigned. (f) The experiments of GAL show that it can obtain very robust classification result with fast accuracy acceleration at the beginning.

Figure 7 reports a group of optimal classification results for different algorithms on real data sets under unlimited parameters. In high-dimensional space, the performance of these AL algorithms is interesting: (a) Random is still stable as dis- cussed in the previous analysis. (b) Margin becomes stable in the high dimension space since the data points are distributed sparsely and no adjacent classes with high density attract the selection process. (c) Hierarchical performs poorly in the high-dimensional space in this group test. After rechecking the algorithm, we find the real reason which leads to this phenomenon is that there is no obvious hierarchical clustering results. Especially for some multi-class data sets, most of the data points are clustered into one class. Then, the algorithm will wrongly label the large class using its label. Wrong clustering results make the algorithm lose capability. (d) TED is still stable in this group test due to its good sampling strategy. (e) Re-active's sensitivity to noises disappears since there are no noises in the *letter* data set. Then, strong classification results are generated. (f) For our GAL algorithm, its performance is still relatively good.

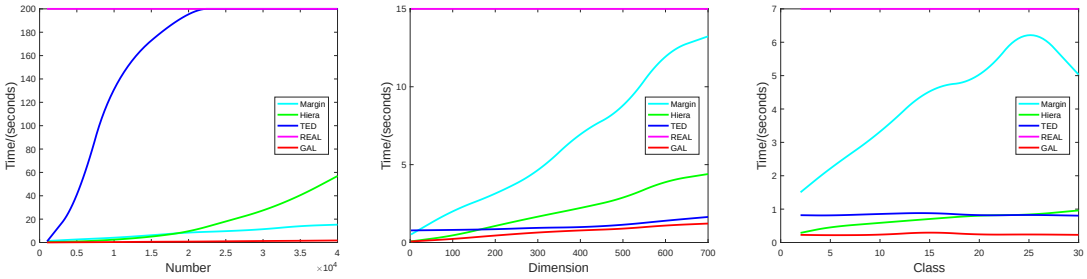
## 7. Discussion

In Section 7.1, we discuss the performance disagreement of different baselines in term of the above experimental results. In Section 7.2, we firstly present the time complexities for these baselines and then organize a group of tests to further analyze their time consumption.

### *7.1. Performance disagreement*

Querying the labels from a group unlabeled instances can drastically improve the current training model. However, how to select the most informative or representative instances from massive unlabeled data is challenging. Generally, random sampling presents a lower bound for the performance of AL sampling. It is a fast method with low time consumption. In the view of theoretical time complexity, its time price is lower than most statistical sampling approaches.

As a typical AL method which use uncertainty evaluation, the adjacent class bias of Margin is firstly discovered by us in the multi-class classification problem and its sensitivity to noise also is amplified. In Hierarchical sampling, the decision whether or not to annotate a cluster subtree with a root node's label is evaluated by a probability function. Even though it returns more labeled data without the help of human expert. A series of problems




(a) Number (b) Dimension (c) Class

Figure 8: The relations between running time and data set number, dimension and class. Re-active is a slow algorithm and its time consumption is longer than that of the other algorithms, so we use a horizontal line to represent it. Also, TED needs a long time to execute when the data number is more than $2 \times 10^4$, thus we only show a part of its real line.

inevitably occurs when the evaluation is misled by unstructured clusters (see the classification result in Jain) or insufficient annotated data, although the established probabilistic hypothesis may be helpful. The experimental optimization of TED reduces the redundant rate of sampling results. The cost is tunning more parameters in kernel space. It also leads to a low robust result in terms of parameters. Instead of the common focus on unlabeled data with informativeness or representativeness, Re-active changes the view into the labeled set. Assigning an unlabeled data with a negative or positive label, the error disagreement (difference) on the labeled set become a key property to reflect the perturbation to the current model. However, this method needs repeatedly visit the candidate data pool with a high cost. Meanwhile, it is suggested to apply in binary classification issue due to a unbearable time complexities if assume any unlabeled data with multi labels.

### *7.2. Time complexities*

In the model-based approaches, the time complexity of training classifiers determines the time consumption of sampling process. Studying the time complexity of SVM is $\mathcal{O}(N^2)$ to $\mathcal{O}(N^3)$, we predict that Margin's time cost will rise to $\mathcal{O}(N_s N_L^2)$ to $\mathcal{O}(N_s N_L^3)$ with a given queries number of $N_s$, where $N_L$ is the number of labeled data. For Hierarchical approach, hierarchical clustering is its main time consumption process that costs $\mathcal{O}(N^2)$. Similarly, calculating the kernel matrix also costs the time price of $\mathcal{O}(N^2)$ in TED. Although Re-active is a novel idea, but it needs to visit whole unlabeled pool to select a data by approximately $N$ times SVM training. It means that the time complexity of one selection will cost $\mathcal{O}(N_L^3)$ to $\mathcal{O}(N_L^4)$ and the time consumption of sampling $N_s$ data points will be $\mathcal{O}(N_s N_L^3)$ to $\mathcal{O}(N_s N_L^4)$. Our GAL approach uses the R-tree to calculate the $k$NN matrix of $\mathcal{X}$ with A time complexity of $\mathcal{O}(Nlog(N))$ at the beginning, then it only uses one parameters to select the sampling set under a certain number.

To analyze their time performance of the above approaches, here we show a group of experiments involved to the running time test on data size, dimension, class number in Figure 8. In this group of test, we synthesize a group Gaussian classes via varying its instance number, dimensions, and class number. Before running these baselines, the parameters settings are $N_s = 100$ , $\lambda = 0.01$ in TED, $\lambda_1 = 0.7$ in GAL. In figure (a), we set Dimension= 2, Class= 2, and vary the Gaussian synthetic data set number from 1000 to 40000. In figure (b), we set Number= 1000, Class= 2 and vary the dimension from 2 to 700. In figure (c) we set Number= 1000, Dimension= 2 and vary the class number from 2 to 30. In the presented time curves, Re-active (REAL) costs very expensive even at the beginning of the sampling. Therefore, we use a horizontal straight line to denote its cost since its real cost already exceed the maximum value of y-axis. Besides it, TED also costs expensive in the first group of test due to the expensive cost in RBF kernel calculation.

Observing the drawn curves in Figure 7(a) to (c), we further conclude: 1) REAL costs very expensive and its complexity scale is far above other algorithms; 2) The cost of TED is also involved with dimensions because the time complexity of SVM is proportional to the dimension of data set; 3) When we synthetic more Gaussian classes, we see the time complexities of TED, REAL, and GAL have slight change. Bu, the time cost of Margin algorithm increases rapidly due SVM algorithm produces more support vectors between any two different classes; 4) Margin algorithm reduces its time cost after we synthetic more than 25 classes since there exists more adjacent classes or overlap classes. The number of the generated support vectors decrease. It is thus the distribution of the classes affect the time cost of Margin; 5) Overall, the time consumption of our proposed GAL is lower than other baselines in terms of our above experimental settings.

## 8. Conclusion

In this paper, we propose a divide-and-conquer idea that analyzes the uncertainty evaluation of AL sampling. Inspired by CVM, we divide the data within one cluster into cluster boundary and core points. Main theoretical contribution in geometric perspective shows 1) cluster boundary points have smaller margin distances to classification hyperplane compared to core points, and 2) training hypotheses based on core points are the subset of hypotheses based on cluster boundary points.

With the theoretical advantages of cluster boundary points, we completely eliminate the dependency on uncertainty evaluation functions by sampling in the cluster boundary points. By training those points, we develop a GAL algorithm based on a knight's tour method. The experiment results demonstrate that the GAL algorithm, which trains the cluster boundary points, outperforms other existing cluster boundary and AL baselines.

## Conflict of Interest

The authors declare that they have no conflict of interest.

## References


Bekasov, A., & Murray, I. (2018). Bayesian adversarial spheres: Bayesian inference and adversarial examples in a noiseless setting. *arXiv preprint arXiv:1811.12335*, .

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, *2*, 27.

Cohn, D., Atlas, L., & Ladner, R. (1994). Improving generalization with active learning. *Machine learning*, *15*, 201–221.

Dasgupta, S., & Hsu, D. (2008). Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning* (pp. 208–215). ACM.

Fey, M., Eric Lenssen, J., Weichert, F., & Müller, H. (2018). Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 869–877).

Guo, Y., Ding, G., Wang, Y., & Jin, X. (2016). Active learning with cross-class knowledge transfer. In *AAAI* (pp. 1624–1630).

Hu, R., Mac Namee, B., & Delany, S. J. (2016). Active learning for text classification with reusability. *Expert Systems with Applications*, *45*, 438–449.

Kang, J., Ryu, K. R., & Kwon, H.-C. (2004). Using cluster-based sampling to select initial training set for active learning in text classification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 384–388). Springer.

Lewis, D. D., & Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 3–12). Springer-Verlag New York, Inc.

Li, H., Shi, Y., Liu, Y., Hauptmann, A. G., & Xiong, Z. (2012). Cross-domain video concept detection: A joint discriminative and generative active learning approach. *Expert Systems with Applications*, *39*, 12220–12228.

Li, X., Geng, P., & Qiu, B. (2015). Clustering boundary detection for high dimensional space based on space inversion and hopkins statistics. *Control Decision*, *30*, 171–175.

Lin, C. H., Mausam, M., & Weld, D. S. (2016). Re-active learning: Active learning with relabeling. In *AAAI* (pp. 1845–1852).

Melville, P., & Mooney, R. J. (2004). Diverse ensembles for active learning. In *Proceedings of the twenty-first international conference on Machine learning* (p. 74). ACM.

Nguyen, H. T., & Smeulders, A. (2004). Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning* (p. 79). ACM.

Qiu, B., & Cao, X. (2016). Clustering boundary detection for high dimensional space based on space inversion and hopkins statistics. *Knowledge-Based Systems*, *98*, 216–225.

Qiu, B., Feng, Y., & Yi, S. J. (2007). Brim: An efficient boundary points detecting algorithm. *Advances in Knowledge Discovery and Data Mining*, .

Roy, N., & McCallum, A. (2001). Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, (pp. 441–448).

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 287–294). ACM.

Tax, D. M., & Duin, R. P. (2004). Support vector data description. *Machine learning*, *54*, 45–66.

Tong, S., & Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of machine learning research*, *2*, 45–66.

Tsang, I.-H., Kwok, J.-Y., & Zurada, J. M. (2006). Generalized core vector machines. *IEEE Transactions on Neural Networks*, *17*, 1126–1140.

Tsang, I. W., Kocsor, A., & Kwok, J. T. (2007). Simpler core vector machines with enclosing balls. In *Proceedings of the 24th international conference on Machine learning* (pp. 911–918). ACM.

Tsang, I. W., Kwok, J. T., & Cheung, P.-M. (2005). Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, *6*, 363–392.

Urner, R., Wulff, S., & Ben-David, S. (2013). Plal: Cluster-based active learning. In *Conference on Learning Theory* (pp. 376–397).

Xia, C., Hsu, W., Lee, M.-L., & Ooi, B. C. (2006). Border: efficient computation of boundary points. *IEEE Transactions on Knowledge and Data Engineering*, *18*, 289–303.

Yu, K., Bi, J., & Tresp, V. (2006). Active learning via transductive experimental design. In *Proceedings of the 23rd international conference on Machine learning* (pp. 1081–1088). ACM.

Zhang, L., Chen, C., Bu, J., Cai, D., He, X., & Huang, T. S. (2011). Active learning based on locally linear reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*, 2026–2038.