A SAFE AND RELIABLE ACTIVE LEARNING METHOD

Anonymous authors
Paper under double-blind review

ABSTRACT

Active learning (AL) aims to alleviate the annotation cost of deep supervised learning. Recently, coverage-aware AL methods obtain state-of-the-art performance across a wide range of budget settings. However, existing approaches often pay limited attention to the purity component of the coverage-purity framework. Explicitly optimizing purity in AL is challenging because of the inherent trade-off between coverage and purity, as well as the lack of an effective mechanism for incorporating purity into the AL process. To address these issues, we introduce the safe zone framework which identifies reliable, high-purity regions within the unlabeled pool and thereby preserves the purity of the total covered area. We further provide theoretical guarantees for the purity achieved by the resulting safe zone. Subsequently, we propose SafeHerding, an efficient purity-aware AL method that prioritizes high-purity samples to improve model performance. The safe zone framework provides the foundation for key components of SafeHerding, including coverage optimization and kernel herding paradigm. In addition, SafeHerding incorporates a novel skipping strategy to manage the intrinsic trade-off between coverage and purity. Experimental results across different learning settings demonstrate that SafeHerding consistently outperforms existing baselines.

1 INTRODUCTION

Deep learning has achieved state-of-the-art performance across a wide range of supervised learning tasks (Noor & Ige, 2025). However, such success often relies on large-scale labeled datasets, whose annotation can impose substantial burdens on human oracles in terms of time and resources. Active learning (AL) (Li et al., 2024) aims to alleviate this annotation cost by developing effective selection strategies that identify the most informative samples for labeling. Existing AL methods generally follow two main sampling principles. The first is uncertainty-based sampling (Nguyen et al., 2022), which estimates the predictive uncertainty of unlabeled samples and prioritizes those about which the model is most uncertain. The second is representation-based sampling (Yehuda et al., 2022; Bae et al., 2024), which selects samples from representative regions of the feature space to better capture the underlying data distribution while reducing redundancy in the labeled set. In addition, several studies (Ash et al., 2020; Parvaneh et al., 2022) combine uncertainty and representation within a unified selection strategy, commonly known as hybrid sampling.

Uncertainty-based AL methods have been shown to perform well in high-budget regimes, where the model is sufficiently trained for its predictive uncertainty to be reliable, but they often struggle in low-budget settings (Hacohen et al., 2022). In contrast, representation-based methods such as ProCover (Yehuda et al., 2022), MaxHerding (Bae et al., 2024) aim to select representative samples from the unlabeled pool and are particularly effective when the annotation budget is limited. These approaches introduce a generalized coverage-purity framework that selects samples to maximize the coverage of the underlying data distribution. Although representation-based sampling achieves strong performance in low-budget regimes, its effectiveness generally decreases as the annotation budget grows. To bridge these two regimes while maintaining robust performance across different budgets, Bae et al. (2025) proposes UHerding, which incorporates uncertainty into the coverage-purity framework. Specifically, UHerding prioritizes representative samples in low-budget settings and gradually shifts its focus toward uncertain samples as the available annotation budget increases. As a result, UHerding achieves state-of-the-art performance across a wide range of budget settings.

Despite their success, existing coverage-aware methods (Yehuda et al., 2022; Bae et al., 2024; 2025) often pay limited attention to the purity component of the coverage-purity framework, which may

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

adversely affect their performance and lead to suboptimal sample selection. On the other hand, explicitly optimizing purity in AL is challenging for several reasons. First, there exists an inherent trade-off between coverage and purity (Yehuda et al., 2022), implying that maximizing coverage alone does not necessarily lead to high-purity regions. In fact, expanding the covered region may increase the likelihood of including samples from different classes, thereby reducing its overall purity. Second, there is currently a lack of an effective framework for directly handling purity in the AL setting. Evaluating and optimizing purity typically requires access to the true labels of all samples within the covered region, which is unavailable for the unlabeled pool in the AL setting. Nevertheless, jointly considering coverage and purity is important for reducing the generalization error and improving the model performance.

In this work, we introduce the safe zone framework which identifies reliable, high-purity regions within the unlabeled pool and thereby preserves the purity of the total covered area. This framework includes a hyperplane-based construction to split the data space into different search regions, followed by a self-correction algorithm that refines these regions into more reliable and pure regions. We also provide theoretical guarantees for the purity achieved by the resulting safe zone. Building on this framework, we propose SafeHerding, an efficient purity-aware AL method that prioritizes high-purity samples to improve model performance. The safe zone framework provides the foundation for key components of SafeHerding, including coverage optimization and kernel herding paradigm. In addition, SafeHerding incorporates a novel skipping strategy to manage the intrinsic trade-off between coverage and purity while also handling trivial safe-region cases. Finally, extensive experiments in both supervised and transfer learning settings demonstrate that SafeHerding consistently outperforms uncertainty-based and representation-based baselines.

2 PRELIMINARIES

We formulate the AL problem in the standard pool-based setting. Let $\mathcal{X}$ denote the entire set of input samples, $\mathcal{Y} = \{1, \ldots, |\mathcal{Y}|\}$ the class label set, and f the true labeling function $f : \mathcal{X} \to \mathcal{Y}$. At the beginning, the data $\mathcal{X}$ are partitioned into a labeled set $\mathcal{L}$ and an unlabeled pool $\mathcal{U} = \mathcal{X} \setminus \mathcal{L}$. In typical AL scenarios, the labeled set is much smaller than the unlabeled pool, i.e. $|\mathcal{L}| \ll |\mathcal{U}|$. The learning process proceeds for T rounds. At each round t, the **sampler** $\mathbf{S}$ selects a subset of points from $\mathcal{U}$ and added to $\mathcal{L}$, thereby providing new training samples for the main **learner** $\hat{f}$.

Sampler. Let $\mathcal{L}_t$ and $\mathcal{U}_t$ denote the labeled set and unlabeled pool at round t, respectively. In round $t+1$, the sampler $\mathbf{S}$ evaluates the samples in $\mathcal{U}_t$ according to its selection criterion, assigning each sample a score that reflects its potential informativeness. Based on these scores, $\mathbf{S}$ chooses $\mathcal{B}_t$ samples $\{x_i\}_{i=1}^{\mathcal{B}_t}$ from $\mathcal{U}_t$ to be labeled and used for training the learner $\hat{f}$, where $\mathcal{B}_t$ is the labeling budget. The labeled and unlabeled sets are then updated as:

$$\mathcal{L}_{t+1} = \mathcal{L}_t \cup \{x_i\}_{i=1}^{\mathcal{B}_t}, \mathcal{U}_{t+1} = \mathcal{U}_t \setminus \{x_i\}_{i=1}^{\mathcal{B}_t}. \tag{1}$$

Learner. The learner $\hat{f}$ is a classification model parameterized by $\hat{\theta}$. The main objective of AL is to train the learner effectively so as to minimize the generalization error with respect to the true labeling function f:

$$\hat{\theta}^* = \arg\min_{\hat{\theta}} \mathbb{E}\big[\hat{f}_{\hat{\theta}}(x) \neq f(x)\big]_{\mathcal{X}} \tag{2}$$

Unlike supervised learning, where the model is trained on a large labeled dataset, active learning operates under a much more constrained setting, in which the available labeled samples $\mathcal{L}_t$ for training the learner $\hat{f}$ is limited.

The coverage-purity framework. Similar to (Yehuda et al., 2022; Bae et al., 2024; 2025), our work is based on the coverage-purity framework. This framework formulates the sampling process in terms of either coverage optimization or purity optimization. Under coverage optimization, the sampler $\mathbf{S}$ seeks to select a sample x whose neighborhood contains as many data points as possible, thereby maximizing the represented portion of the data space. Meanwhile, under purity optimization, the sampler $\mathbf{S}$ favors a sample x whose neighborhood is composed predominantly of samples sharing the same label as x, thus focusing on reliable samples that better represent class-specific regions. More details about the coverage-purity framework are shown in Appendix C.

3 SAFE ZONE FRAMEWORK

To integrate the purity in active learning setting, we introduce a safe zone framework to capture high-purity samples which we refer to as **safe samples**. At the core of our framework is a hyperplane-based construction that partitions the entire data space $\mathcal{X}$ into M distinct search regions $\{(\beta_i)\}_{i=1}^{M}$. Under active learning setting, all search regions $\{(\beta_i)\}_{i=1}^{M}$ are considered as discrete in this work. We then develop a self-correction algorithm to progressively refine these initial search regions into more reliable **safe regions**. The union of these safe regions forms the **safe zone** $\Omega_{\text{safe}} = \bigcup_i (\beta_i)$. By restricting sample selection to the safe zone, the sampler $\mathbf{S}$ can avoid potentially impure regions and select high-purity samples with theoretical guarantees.

3.1 HYPERPLANE-BASED CONSTRUCTION

To establish the initial search regions $\{(\beta_i)\}_{i=1}^{M}$, we first select a reliable centroid c_i for each region (β_i). For this purpose, we leverage the labeled set $\mathcal{L}_t$ collected during the active learning process, as detailed in Appendix H. In the early rounds, the sampler $\mathbf{S}$ explores the full data space and acquires a set of labeled samples, which are then treated as centroid candidates for region construction. This strategy is adopted for two reasons. First, the true label $f(c_i)$ of each centroid is required because it acts as the reference label for region (β_i), allowing us to evaluate whether samples assigned to that region are consistent with the centroid and thus estimate its purity. Second, since annotation is expensive in active learning, reusing already labeled samples avoids introducing any additional labeling cost beyond the standard AL procedure.

Once the set of centroids $\{c_i\}_{i=1}^{M}$ is obtained, where each $c_i \in \mathbb{R}^{d\times 1}$, we construct $M-1$ boundary hyperplanes $\{(\alpha_{i,j})\}_{j\neq i}$ for each search region (β_i). Each hyperplane $(\alpha_{i,j})$ defines the boundary between the regions associated with centroids c_i and c_j. Specifically, $(\alpha_{i,j})$ is characterized by a normal vector $\overrightarrow{n_{i,j}}$ and passes through the midpoint $a_{i,j}$ of two centroids c_i and c_j:

$$a_{i,j} = \frac{c^i + c^j}{2} \in \mathbb{R}^{d\times 1}, \overrightarrow{n_{i,j}} = \frac{(c^j - c^i)}{\|c^j - c^i\|} \in \mathbb{R}^{d\times 1}, \tag{3}$$

$$(\alpha_{i,j}) : (\overrightarrow{n_{i,j}})^T \odot (x - a_{i,j}) = 0, \tag{4}$$

where $\odot$ denotes the inner product in Euclidean space. The search region (β_i) is then defined as the set of all points x that lie on the same side as the centroid c_i with respect to every separating hyperplane $(\alpha_{i,j})$:

$$(\beta_i) : \{x \in \mathbb{R}^{d\times 1} | \text{sign}((\overrightarrow{n_{i,j}})^T \odot (x - a_{i,j})) = \text{sign}((\overrightarrow{n_{i,j}})^T \odot (c_i - a_{i,j})) \forall j \neq i\}. \tag{5}$$

Determining the search region (β_i) for each unlabeled sample $x \in \mathcal{U}_t$ can be computationally expensive and memory-inefficient if the assignment is performed directly through sign evaluation of the inner products defined in Eq. (5). This sign-based assignment requires a complexity of $O(|\mathcal{U}_t|M^2 d)$ where M is the number of search regions, d is the feature dimension. To reduce this cost, we instead assign each sample to its corresponding search region by identifying its nearest centroid, which leads to a more efficient computation of $O(|\mathcal{U}_t|Md)$. We further provide **Proposition 1** to show that this nearest-centroid assignment is consistent with the definition of the search regions.

Proposition 1. *Given the unlabeled pool $\mathcal{U}_t$, and a set of centroids $\{c_i\}_{i=1}^{M}$, any sample $x \in \mathcal{U}_t$ belongs to the search region (β_i) if and only if c_i is the nearest centroid to x.*

$$x \in (\beta_i) \iff i = \arg\min_{j\in\{1,\ldots,M\}} \|x - c_j\|.$$

It is worth noting that the hyperplane-based construction is performed only once at the beginning, whereas the self-correction procedure in Section 3.2 is applied iteratively to refine search regions. Consequently, the nearest-centroid assignment property holds only for the initial search regions and is not necessarily preserved after refinement, as self-correction removes incorrectly assigned samples to improve purity. This modification is an intentional part of the refinement process and does not compromise the validity of the proposed framework.

3.2 SELF-CORRECTION ALGORITHM

Fig. 1 demonstrates the overall procedure of our proposed self-correction algorithm. After applying the hyperplane-based construction in Section 3.1, we obtain the initial search regions. However, since

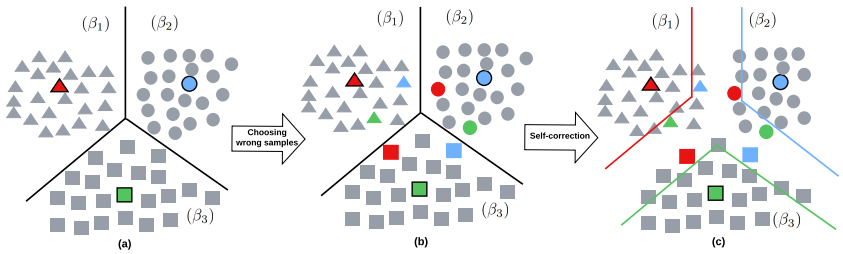

Figure 1: Illustration of the proposed self-correction algorithm. The shapes indicate the assigned search regions, while the colors represent the ground-truth class labels. (a) Initial search regions obtained through the hyperplane-based construction. (b) The sampler $\mathbf{S}$ selects several incorrectly assigned samples. (c) The self-correction algorithm adjusts the separating hyperplanes to exclude the incorrectly assigned samples and refine the search regions.

the ground-truth labels of the unlabeled samples within these regions are unavailable, the purity of the initial search regions cannot be guaranteed. Consequently, when the sampler $\mathbf{S}$ selects samples from these regions, it may query incorrectly assigned samples x whose region labels are inconsistent with their ground-truth class labels, i.e. $f(x) \neq f(c_i)$. To address this issue, we introduce a self-correction algorithm that adjusts the separating hyperplanes to exclude the incorrectly assigned samples, thereby progressively refining the search regions and improving their purity.

Consider a search region (β_i) bounded by the set of separating hyperplanes $\{(\alpha_{i,j})\}_{i\neq j}$ defined in Eq. (3) and Eq. (4). To enable an efficient vectorized implementation, we collect the parameters of these $M-1$ hyperplanes into a *weight matrix* W_i and a *bias vector* b_i, defined as:

$$W_i = [(\overrightarrow{n_{i,j}})^T]_{j\neq i} \in \mathbb{R}^{(M-1)\times d}, \tag{6}$$

$$b_i = [-(\overrightarrow{n_{i,j}})^T \odot a_{i,j}]_{j\neq i} \in \mathbb{R}^{(M-1)\times 1}. \tag{7}$$

Each row of W_i therefore corresponds to the transposed normal vector of one boundary hyperplane, while the corresponding element of b_i specifies its offset from the origin. Consequently, the j-th boundary of (β_i) can be written in the affine form:

$$(\alpha_{i,j}) : (\overrightarrow{n_{i,j}})^T x + b_{i,j} = 0. \tag{8}$$

Note that the bias terms are not required to be symmetric, i.e., $b_{i,j} \neq b_{j,i}$. This asymmetry allows the corresponding hyperplane boundaries to be updated independently for each search region, providing greater flexibility during the self-correction process. Suppose that, during the active learning process, the sampler $\mathbf{S}$ queries K wrong samples $X_K = [x_1, \ldots, x_K]^T \in \mathbb{R}^{K\times d}$, where each sample $x_k \in (\beta_i)$ and $f(x_k) \neq f(c_i)$. Their signed distances to all boundary hyperplanes of (β_i) can be computed simultaneously as:

$$\mathcal{D} = W_i (X_K)^T + b_i \in \mathbb{R}^{(M-1)\times K}, \tag{9}$$

where the element $\mathcal{D}_{j,k}$ is the signed distance from x_k to the hyperplane $(\alpha_{i,j})$. For each incorrectly assigned sample x_k, we identify its nearest boundary hyperplane by:

$$j^* = \arg\min_j |\mathcal{D}_{j,k}|. \tag{10}$$

We then correct the corresponding boundary by translating (α_{i,j^*}) toward x_k. Since the normal vector is kept fixed, this translation can be performed by modifying only its bias term. In particular, we update:

$$b_{i,j^*} \leftarrow b_{i,j^*} - \mathcal{D}_{j^*,k} \text{ if } |\mathcal{D}_{j^*,k}| < \mathbf{d}_i = |(\overrightarrow{n_{i,j^*}})^T c_i + b_{i,j^*}|, \tag{11}$$

where $\mathbf{d}_i$ is the Euclidean distance from the centroid c_i to the hyperplane (α_{i,j^*}).

Selecting the nearest boundary in Eq. (10) yields the smallest translation required to exclude an incorrectly assigned sample. This strategy avoids unnecessarily large changes to the search region, thereby retaining as many correctly assigned samples as possible. Meanwhile, the condition in Eq. (11) ensures that the centroid remains within its corresponding search region while maintaining robustness against noisy samples. In the worst-case scenario, where all queried samples are incorrectly

assigned and approximately $K = \mathcal{B}/M$ samples are associated with each search region, the overall computational complexity of the self-correction algorithm is $O(KM^2d) = O(\mathcal{B}Md)$, where $\mathcal{B}$ is the labeling budget, M is the number of search regions, and d is the feature dimension.

To conclude this section, we present **Theorem 1**, which establishes the finite convergence of the proposed self-correction algorithm and provides a theoretical guarantee for its reliability. For clarity, we state a concise version of **Theorem 1** below, while its complete formulation and detailed proof are provided in Appendix D. Further discussions about **Theorem 1** are presented in Appendix H.

Theorem 1 (Finite Convergence of Self-Correction)**.** *Consider a search region $(\beta_i)^{(0)}$ associated with centroid c_i and class label $y_i = f(c_i)$. The search region is bounded by $M-1$ hyperplanes $\{(\alpha_{i,j})\}_{i\neq j}$ defined in Eq. (3) and Eq. (4). Let s denote the self-correction step, and assume that there exists a non-empty fixed **pure region** $(\beta_i)^* \subseteq (\beta_i)^{(0)}$ and $c_i \in (\beta_i)^*$. Then the self-correction procedure satisfies the following properties:*

1. *The sequence of search regions is monotonically shrinking: $(\beta_i)^{(s+1)} \subseteq (\beta_i)^{(s)}$.*
2. *The pure region is preserved throughout the correction process: $(\beta_i)^* \subseteq (\beta_i)^{(s)}$.*
3. *The centroid is preserved: $c_i \in (\beta_i)^{(s)}$ in every self-correction step s.*
4. *Every wrong sample is permanently excluded from the search region if it satisfies the update condition in Eq. (11).*
5. *If the active learning pool is finite, the self-correction algorithm terminates after a finite number of updates.*

*Consequently, after a finite number of correction steps, the bias vector b_i stops changing and the resulting search region $(\beta_i)^{(S)} = (\beta_i)^* \cup \mathcal{N}_i^{(S)}$ where $\mathcal{N}_i^{(S)}$ is the set of incorrectly assigned samples that violate the condition in Eq. (11). The union of all resulting search regions $\Omega_{safe} = \bigcup_i (\beta_i)^{(S)}$ is a complete **safe zone**.*

The set $\mathcal{N}_i^{(S)}$ represents noisy samples, which rarely occurs in practice. Hence, its cardinality $|\mathcal{N}_i^{(S)}|$ is expected to remain small, resulting in high purity within the safe region $(\beta_i)^{(S)}$. Note that in **Theorem 1**, the size of the pure region $(\beta_i)^*$ may naturally become very small and can even converge toward one. In such cases, the resulting region $(\beta_i)^{(S)}$ covers only a limited portion of the data space, making it insufficiently informative for sample selection and potentially restricting the model's learning capacity. We refer to these cases as trivial safe-region cases, which may lead to redundant sample selection in SafeHerding. To mitigate this issue, we introduce the skipping strategy in Section 4, which is designed to detect and avoid such limited safe regions.

4 SafeHerding

To overcome the limitations of coverage-aware AL methods, we propose SafeHerding, an efficient purity-aware active learning approach. Building upon the safe zone framework in Section 3, SafeHerding extends the conventional coverage optimization and kernel herding paradigm by restricting sample selection to reliable regions, thereby explicitly incorporating purity into the active learning process. Furthermore, we introduce a novel agreement-based skipping scheme to overcome the inherent trade-off and the trivial safe-region cases. This mechanism adaptively relaxes the safe-zone constraint when appropriate, preventing redundant sampling and improving the performance of the learner $\hat{f}$ in later rounds.

4.1 Safe Coverage

UHerding (Bae et al., 2025) introduces the notion of *uncertainty coverage*, which quantifies the amount of predictive uncertainty represented by a selected set of samples and serves as the optimization objective for its sample selection strategy. By incorporating uncertainty into the coverage formulation, UHerding can effectively adapt to different annotation-budget regimes, ranging from low to high budgets, and consequently achieves strong performance against existing baselines. We recall the definition of uncertainty coverage as follows:

Definition 1 (***Uncertainty Coverage***). *For any subset $\mathcal{Z} \subset \mathcal{X}$, a nonnegative-value kernel σ_γ, a feature extractor g, and a nonnegative-valued uncertainty function U, uncertainty coverage is defined and empirically estimated as:*

$$UC_{\sigma_\gamma}(\mathcal{Z}) = \mathbb{E}_{\mathbf{x}}\left[U(\mathbf{x};\hat{f}) \max_{\mathbf{x}' \in \mathcal{Z}} \sigma_\gamma(\mathbf{x},\mathbf{x}';g)\right] \approx \frac{1}{N}\sum_{n=1}^{N} U(\mathbf{x}_n;\hat{f}) \max_{\mathbf{x}' \in \mathcal{Z}} \sigma_\gamma(\mathbf{x}_n,\mathbf{x}';g) = \widehat{UC}_{\sigma_\gamma}(\mathcal{Z}).$$

UHerding employs the kernel:

$$\sigma_\gamma(\mathbf{x},\mathbf{x}';g) = \exp(-\|(g(\mathbf{x}) - g(\mathbf{x}'))/\gamma\|^2), \tag{12}$$

where γ is an adaptive radius parameter that controls the locality of the kernel. By adjusting γ, UHerding can transition between low-budget and high-budget regimes while maintaining robust sample selection performance (Bae et al., 2025).

Building upon this formulation, we incorporate the safe zone framework introduced in Section 3 into uncertainty coverage. Specifically, we define a *safe kernel* $\tilde{\sigma}_\gamma$ as:

$$\tilde{\sigma}_\gamma(\mathbf{x},\mathbf{x}';g) = \begin{cases} \sigma_\gamma(\mathbf{x},\mathbf{x}';g) & \text{if } \mathbf{x} \text{ and } \mathbf{x}' \in \mathcal{L}_t \cup \Omega_{\text{safe}}, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Unlike the original kernel in Eq. (12), which measures similarity over the entire data space, the proposed kernel in Eq. (13) only preserves interactions between samples belonging to the labeled set or the safe zone. Substituting Eq. (13) into **Definition 1** leads to our notion of *safe coverage* SC, defined as:

$$SC(\mathcal{Z}) = UC_{\tilde{\sigma}_\gamma}(\mathcal{Z}),\ \widehat{SC}(\mathcal{Z}) = \widehat{UC}_{\tilde{\sigma}_\gamma}(\mathcal{Z}). \tag{14}$$

Safe coverage can therefore be viewed as a constrained form of uncertainty coverage that explicitly incorporates the safe zone. Intuitively, it measures how effectively the selected set $\mathcal{Z}$ covers uncertain samples within reliable regions while suppressing the influence of potentially impure samples. Based on this objective, we propose SafeHerding, which greedily selects the sample that maximizes the empirical safe coverage:

$$\mathbf{x}^* \in \arg\max_{\tilde{x} \in \mathcal{U}} \widehat{SC}(\mathcal{L} \cup \{\tilde{x}\}) = \arg\max_{\tilde{x} \in \mathcal{U}} \left(\frac{1}{N}\sum_{n=1}^{N} U(\mathbf{x}_n;\hat{f}) \max_{\mathbf{x}' \in \mathcal{L} \cup \{\tilde{x}\}} \tilde{\sigma}_\gamma(\mathbf{x}_n,\mathbf{x}';g)\right). \tag{15}$$

The restriction imposed by the safe kernel $\tilde{\sigma}_\gamma$ inevitably reduces the maximum coverage that can be achieved compared with the unconstrained uncertainty coverage of UHerding. This relationship is formalized in the following proposition.

Proposition 2. *For any subset $\mathcal{Z} \subseteq \mathcal{X}$, the empirical safe coverage is upper bounded by the corresponding empirical uncertainty coverage:*

$$\widehat{SC}(\mathcal{Z}) \leq \widehat{UC}_{\sigma_\gamma}(\mathcal{Z}).$$

Although **Proposition 2** shows that enforcing the safe zone constraint may sacrifice a portion of the achievable coverage, this restriction is intentional. The safe kernel directs the optimization in Eq. (15) toward Ω_{safe} and eliminates contributions from potentially impure regions, encouraging SafeHerding to select samples that produce a purer total covered area than MaxHerding and UHerding.

4.2 Agreement-based Skipping

In addition to the trivial safe-region issue discussed in Section 3.2, **Theorem 1** and **Proposition 2**, imply that the gap between safe coverage and the original uncertainty coverage may increase over time. As a result, SafeHerding may become overly conservative, limiting its exploration capability and potentially degrading learner performance in later rounds. These observations motivate an adaptive transition from purity-aware selection toward broader coverage-aware exploration. To enable such a transition, we introduce an *agreement score* ϕ_i, which measures the consistency between the predictions of the learner $\hat{f}$ and the reference label associated with search region (β_i).

Definition 2 (***Agreement Score***). *Given a search region (β_i) with reference label $y_i = f(c_i)$, the agreement score ϕ_i is defined as:*

$$\phi_i = \frac{1}{|(\beta_i)|} \sum_{x \in (\beta_i)} \mathbb{1}\Big[\hat{f}(x) = f(c_i)\Big]. \tag{16}$$

The agreement score satisfies $\phi_i \in [0, 1]$. A larger value of ϕ_i indicates that a larger proportion of samples within (β_i) are predicted by the learner as belonging to the region label $f(c_i)$. Therefore, ϕ_i provides an estimate of how consistently the learner interprets the semantic structure of the corresponding search region. To characterize its relationship with prediction error, we further define the purity π_i of region (β_i) with respect to its reference label as:

$$\pi_i = \frac{1}{|(\beta_i)|} \sum_{x \in (\beta_i)} \mathbb{1}[f(x) = f(c_i)]. \tag{17}$$

We also define the empirical classification error ε_i within (β_i) as:

$$\varepsilon_i = \frac{1}{|(\beta_i)|} \sum_{x \in (\beta_i)} \mathbb{1}\Big[\hat{f}(x) \neq f(x)\Big]. \tag{18}$$

The following proposition establishes the relationship among agreement, purity, and prediction error.

Proposition 3. *For any search region (β_i), the empirical classification error satisfies*

$$\varepsilon_i \leq (1 - \phi_i) + (1 - \pi_i).$$

In particular, if (β_i) is pure, i.e., $\pi_i = 1$, then $\varepsilon_i \leq 1 - \phi_i$. Therefore, $\phi_i \to 1 \Longrightarrow \varepsilon_i \to 0$ for a pure search region.

Proposition 3 provides a theoretical interpretation of the agreement score. In a high-purity search region, a large π_i indicates that the learner has already captured the dominant semantic information of that region. In the trivial safe-region case, the purity π_i may attain its maximum value of 1 when the search region (β_i) contains only the centroid. Consequently, the agreement score ϕ_i also approaches 1, while the corresponding prediction error ε_i becomes negligible, indicating that the region contributes little additional information to the learner. Therefore, once a search region (β_i) satisfies the following agreement criterion, it is removed from the safe-zone constraint in SafeHerding:

$$\phi_i \geq \lambda, \tag{19}$$

where $\lambda \in [0, 1]$ is a *local threshold* hyperparameter. When Eq. (19) holds, we regard the learner as sufficiently familiar with the semantic structure of (β_i) and exclude this region from further purity-constrained selection. Accordingly, the safe zone is updated as:

$$\Omega_{\text{safe}} \leftarrow \Omega_{\text{safe}} \setminus (\beta_i). \tag{20}$$

This region-wise skipping mechanism gradually reduces the influence of the safe-zone constraint as learning progresses. To enable a faster global transition from purity-aware selection to coverage-aware exploration, we further introduce a *global threshold* $\Lambda \in [0, 1]$ based on the average agreement score $\bar{\phi}$ across all M search regions. When

$$\bar{\phi} \geq \Lambda, \tag{21}$$

we consider the learner to have acquired sufficient knowledge of the overall search space. At this stage, SafeHerding completely removes the safe-zone constraint and switches from optimizing safe coverage to optimizing the original uncertainty coverage defined in **Definition 1**.

4.3 Overall Workflow

For completeness and reproducibility, the detailed workflow and pseudocode of SafeHerding are provided in Appendix H. Overall, SafeHerding operates through three main stages. It initially adopts uncertainty coverage until sufficient labeled samples are available to construct the safe zone. It then performs purity-aware selection by maximizing safe coverage and progressively refining the safe zone through self-correction. As the learner becomes more reliable, the agreement-based skipping mechanism gradually relaxes the safe-zone constraint and eventually transitions SafeHerding back to uncertainty coverage, enabling broader exploration in later rounds.

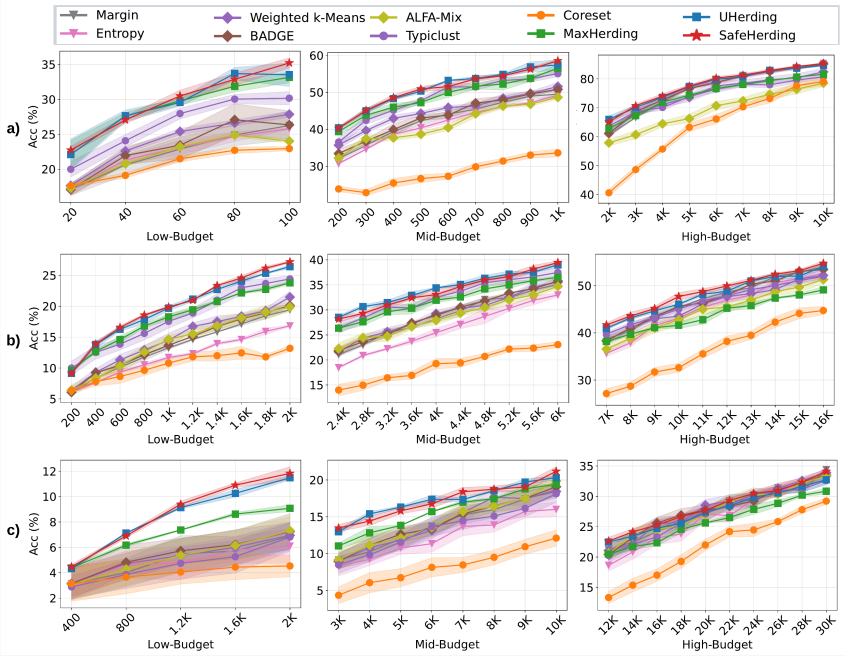

Figure 2: Supervised learning results in terms of Accuracy (%). Results are organized by row for three datasets: (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

5 Experiments

We evaluate the performance of our proposed SafeHerding on supervised learning and transfer learning settings, and conduct a comprehensive comparison against a wide rage of state-of-the-art AL baselines. Details on the benchmark datasets and baseline methods used in our experiments are provided in Appendix G. Our implementation is available at the GitHub repository.

5.1 Supervised Learning Result

Setup. In the supervised learning setting, we evaluate the proposed SafeHerding method against several baselines on three benchmark datasets: CIFAR10, CIFAR100, and TinyImageNet. All experiments are conducted over three independent runs using a ResNet-18 backbone (He et al., 2016) with random initialization. We use the self-supervised feature extractor SimCLR (Chen et al., 2020) in this experiment. For SafeHerding, the number of clusters M is set to 40, 200, and 400 for CIFAR10, CIFAR100, and TinyImageNet, respectively. The local threshold λ and global threshold Λ are fixed to 0.9 and 0.8, respectively, across all datasets.

Result. Figure 2 reports the image classification performance measured by accuracy. Across the three datasets, many baseline methods show noticeable sensitivity to the annotation budget. On TinyImageNet, the representation-based MaxHerding method performs competitively under limited budgets, ranking third in both the low- and medium-budget settings, but falls to near the bottom under the high-budget setting. Conversely, uncertainty-based approaches, including Margin and Entropy, become more competitive as the annotation budget increases and achieve their strongest results in the high-budget regime. In comparison, UHerding and SafeHerding exhibit more stable performance across different budget levels. Moreover, SafeHerding improves upon UHerding in most experimental settings. In particular, on CIFAR10 under the low-budget regime, SafeHerding reaches an average accuracy of 35.27%, exceeding UHerding, the second-best method, by 3.19%. On TinyImageNet under the high-budget regime, SafeHerding also achieves a 4.18% improvement over UHerding. The superior performance of SafeHerding can be attributed to its explicit optimization of purity through the safe zone framework. To better understand the performance gains of SafeHerding over MaxHerding and UHerding, we further examine their sampling behavior under the coverage-purity framework in Appendices F.1 and F.2.

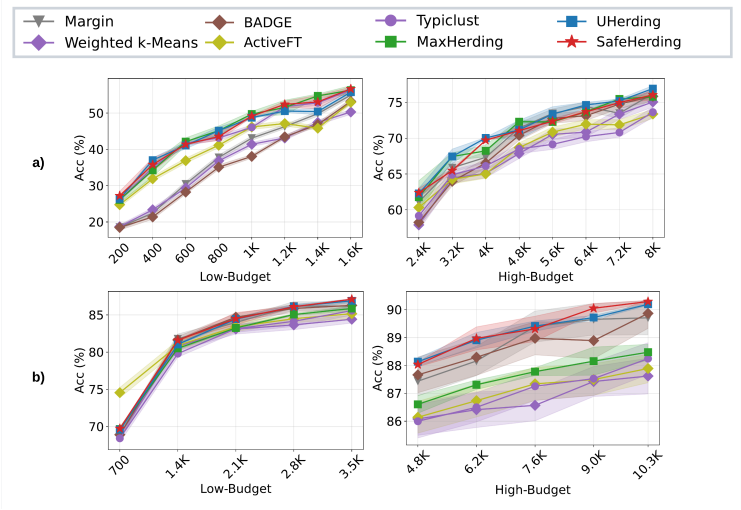

Figure 3: Transfer learning results in terms of Accuracy (%). Results are organized by row for two datasets: (a) CIFAR100 and (b) DomainNet.

5.2 Transfer Learning Result

Setup. For the transfer learning experiment, we validate SafeHerding alongside other AL methods on CIFAR100 and DomainNet. We replace the backbone with a pretrained DeiT model (Touvron et al., 2021). Beside SimCLR, DINO (Caron et al., 2021) is employed as the feature extractor for DomainNet. Similar to the supervised learning experiment, the annotation budgets are manually divided into low- and high-budget regimes. For SafeHerding, the local and global thresholds, λ and Λ, are fixed at 0.9 and 0.8, respectively. The number of clusters M is set to 200 for CIFAR100 and 690 for DomainNet.

Result. Figure 3 presents the transfer learning results. Overall, SafeHerding maintains robust performance across all evaluated settings. On CIFAR100, SafeHerding outperforms UHerding by 1.64% in the low-budget regime, while trailing UHerding by 1.15% in the high-budget regime. On DomainNet, SafeHerding remains highly competitive with UHerding, the strongest baseline, under both low- and high-budget settings.

5.3 Ablation Study

To provide deeper insight into the capacity of SafeHerding, we conduct ablation studies, which are deferred to Appendix F due to space limitations. Appendix F.1 examines the coverage and purity achieved by different methods, while Appendix F.2 further analyzes the underlying factors that lead to these differences. Appendix F.3 investigates the performance gains of SafeHerding and competing approaches relative to Random sampling. Appendix F.4 compares the computational complexity of the three herding-based methods. Appendix F.5 evaluates the impacts of key components in SafeHerding to overall performance, including self-correction algorithm and agreement-based skipping mechanism. Finally, Appendix F.6 assesses the sensitivity of SafeHerding to its key hyperparameters.

6 Conclusion

In this paper, we introduce the safe zone framework which identifies reliable, high-purity regions within the unlabeled pool to preserve the purity of the total covered area. We further establish theoretical guarantees on the purity of the resulting safe zones. Based on our framework, we successfully develop SafeHerding, an efficient purity-aware AL method that achieves promising results across a range of active learning tasks. Despite the extra computational cost, experimental results on real-world datasets demonstrate that SafeHerding outperforms existing AL methods in most evaluated setting.

AI USE STATEMENT

In this work, we have not used generative AI tools for developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, assisting with translation, and interpreting results. Other tasks including generating synthetic data sets, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, are not applicable to this work. Additionally, we used generative AI tools for editing a research paper to improve readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryghZJBKPS.

Wonho Bae, Junhyug Noh, and Danica J Sutherland. Generalized coverage for more robust low-budget active learning. In *European conference on computer vision*, pp. 318–334. Springer, 2024.

Wonho Bae, Danica Sutherland, and Gabriel Oliveira. Uncertainty herding: One active learning method for all label budgets. In *International Conference on Learning Representations*, volume 2025, pp. 788–805, 2025.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pp. 9650–9660, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pp. 109–116, 2010.

Guy Hacohen, Avihu Dekel, and Daphna Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8160–8195. PMLR, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, Ontario, 2009.

Dongyuan Li, Zhen Wang, Yankai Chen, Renhe Jiang, Weiping Ding, and Manabu Okumura. A survey on deep active learning: Recent advances and new frontiers. *IEEE Transactions on Neural Networks and Learning Systems*, 36(4):5879–5899, 2024.

Vu-Linh Nguyen, Mohammad Hossein Shaker, and Eyke Hüllermeier. How to measure uncertainty in uncertainty sampling for active learning. *Machine Learning*, 111(1):89–122, 2022.

Mohd Halim Mohd Noor and Ayokunle Olalekan Ige. A survey on state-of-the-art deep learning applications and challenges. *Engineering Applications of Artificial Intelligence*, 159:111225, 2025.

Amin Parvaneh, Ehsan Abbasnejad, Damien Teney, Gholamreza Reza Haffari, Anton Van Den Hengel, and Javen Qinfeng Shi. Active learning by feature mixing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12237–12246, 2022.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1406–1415, 2019.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In Frank Hoffmann, David J. Hand, Niall Adams, Douglas Fisher, and Gabriela Guimaraes (eds.), *Advances in Intelligent Data Analysis*, pp. 309–318, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44816-7.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

Dan Wang and Yi Shang. A new active labeling method for deep learning. *2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 112–119, 2014. doi: 10.1109/IJCNN.2014.6889457.

Jiayu Wu, Qixiang Zhang, and Guoxia Xu. Tiny imagenet challenge. Technical report, Stanford CS231N, 2017.

Yichen Xie, Han Lu, Junchi Yan, Xiaokang Yang, Masayoshi Tomizuka, and Wei Zhan. Active finetuning: Exploiting annotation budget in the pretraining-finetuning paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23715–23724, June 2023.

Ofer Yehuda, Avihu Dekel, Guy Hacohen, and Daphna Weinshall. Active learning through a covering lens. *Advances in Neural Information Processing Systems*, 35:22354–22367, 2022.

Fedor Zhdanov. Diverse mini-batch active learning, 2019. URL https://arxiv.org/abs/1901.05954.

TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

A NOTATION

Table 1 summarizes all notations mainly used in this work and their corresponding explanations.

Table 1: Main Notations

Notations	Descriptions
$\mathcal{X}$	The entire set of input samples
$\mathcal{Y}$	The class label set
f	The true labeling function
$\mathcal{L}$, $\mathcal{L}_t$	The set of labeled samples (at t-th round)
$\mathcal{U}, \mathcal{U}_t$	The set of unlabeled samples (at t-th round)
U	The uncertainty function
T	The total active learning rounds
t	The t-th active learning round
A^T	The transpose of a matrix or vector A
$\mathbf{S}$	The sampler
$\hat{f}$	The learner
g	The feature extractor
$\mathcal{B}_t$	The annotation budget at t-th round
x, $\mathbf{x}$	A single sample
Ω_{safe}	The safe zone
(β_i)	The i-th search region
M	The total number of search regions
c_i	The centroid of the search region (β_i)
$(\alpha_{i,j})$	The boundary hyperplane between c_i and c_j
$a_{i,j}$	The midpoint between c_i and c_j
$\overrightarrow{n_{i,j}}$	The normal vector between c_i and c_j
$\mathbb{R}$	The Euclidean space
d	The embedding dimension
$d(x, y)$	The Euclidean distance between two points x and y
$\mathcal{D}$	The signed distance matrix
W	The weight matrix
b	The bias vector
σ ,$\tilde{\sigma}$	The kernel function
ϕ_i	The agreement score of region (β_i)
π_i	The purity of region (β_i)
ε_i	The prediction error within (β_i)
λ	The local agreement threshold
Λ	The global agreement threshold
P	The probability
Imp	The impurity

B RELATED WORKS

B.1 BUDGET REGIME DEFINITION

Active Learning (AL) (Li et al., 2024) aims to select a subset of unlabeled samples for annotation while maintaining competitive performance with fully supervised learning. In the pool-based setting, existing studies commonly follow one of two annotation-budget protocols. The first protocol (Parvaneh et al., 2022; Sener & Savarese, 2018; Ash et al., 2020) considers a fixed annotation budget throughout the study, assessing the effectiveness of the proposed method under a specific budget constraint. For example, ALFA-Mix (Parvaneh et al., 2022) identifies informative samples based

on uncertainty induced by feature interpolation, while Coreset (Sener & Savarese, 2018) primarily exploit the representation space to select diverse and representative samples. BADGE (Ash et al., 2020) combines uncertainty and diversity through gradient embeddings.

The second protocol (Hacohen et al., 2022; Yehuda et al., 2022) categorizes the annotation budget into three regimes: low-, mid-, and high- budgets, and evaluates the method across these different budget regimes. The definitions of low-budget, mid-budget, and high-budget regimes in active learning are not standardized, and their corresponding budget ranges vary across studies and datasets (Hacohen et al., 2022; Yehuda et al., 2022; Bae et al., 2025). Consequently, the relative effectiveness of existing AL strategies can vary considerably depending on the available annotation budget.

B.2 Impact of Budget Regime on Active Learning Performance

Uncertainty-based AL methods are generally more effective in high-budget regimes, where the learner has observed sufficient labeled data for its uncertainty estimates to become more reliable. In most cases, uncertainty sampling is derived from the model's predictive probabilities (Li et al., 2024). Confidence-based sampling selects samples for which the model has the lowest prediction confidence. Confidence-based sampling selects instances for which the model assigns the lowest prediction confidence. Margin-based sampling (Scheffer et al., 2001) prioritizes samples with the smallest difference between the probabilities of the two most likely classes, while entropy-based sampling (Wang & Shang, 2014) favors samples with the highest predictive entropy. Weighted Entropy (Zhdanov, 2019) further incorporates representation information by selecting unlabeled samples with sufficiently large margins that are closest to the centroids of weighted k-means clusters.

However, the effectiveness of uncertainty-based sampling generally declines as the annotation budget becomes smaller, with the degradation being especially pronounced in the low-budget regime. In contrast, recent studies have demonstrated that representation-based strategies can provide stronger performance under both low- and medium-budget settings (Hacohen et al., 2022; Yehuda et al., 2022; Bae et al., 2024). TypiClust (Hacohen et al., 2022) was among the first works to highlight the performance transition of active learning methods across different annotation budgets and to introduce a representation-based strategy tailored to the low-budget regime. ProbCover (Yehuda et al., 2022) further studies this problem from the perspectives of coverage and purity, proposing a method that maximizes the coverage of the selected subset within a predefined radius. Although ProbCover achieves promising improvements over TypiClust, its performance is sensitive to the choice of radius. To alleviate this limitation, MaxHerding (Bae et al., 2024) introduces a kernel-herding-based approach that extends the theoretical foundation of ProbCover. Its goal is to derive a more general and stable upper bound on the generalization error within the coverage-purity framework.

Furthermore, UHerding (Bae et al., 2025) incorporates model uncertainty into the kernel herding framework, enabling the selection strategy to adapt to different annotation budgets and achieve competitive performance from low-budget to high-budget regimes. Despite their effectiveness, these coverage-based samplings (Yehuda et al., 2022; Bae et al., 2024; 2025) primarily focus on approximating the underlying data distribution through representative samples, without explicitly enforcing a purity constraint. This becomes a critical issue because an effective selected subset should not only cover the data distribution broadly, but also avoid regions containing heterogeneous or potentially impure samples.

C The Coverage-Purity Framework

The coverage-purity framework was first introduced to active learning by ProbCover (Yehuda et al., 2022) for informative sample selection. Suppose that at round t, we have selected a labeled subset $\{x_i\}_{i=1}^{|\mathcal{L}_t|}$, and each sample x_i induces a local neighborhood region, modeled as a ball $B(x_i, r)$ centered at x_i with radius r. The coverage is then defined as follows:

Definition 3 (***Coverage***). *Given a set of neighborhoods $\{B(x_i, r)\}_{i=1}^{|\mathcal{L}_t|}$, the union $\mathcal{C}(\mathcal{L}_t, r) = \bigcup_{i=1}^{|\mathcal{L}_t|} B(x_i, r)$ defines the **total covered region**, and the **coverage** is the probability mass of this region over the data space:*

$$P(\mathcal{C}) = P_{x\sim\mathcal{X}}(x \in \mathcal{C}(\mathcal{L}_t, r)). \tag{22}$$

Coverage measures how well the selected samples represent the underlying data distribution, such that, under local label consistency, nearby unlabeled samples can be represented by the selected labeled samples. Accordingly, existing coverage-aware active learning methods (Yehuda et al., 2022; Bae et al., 2024; 2025) aim to identify a labeled subset $\mathcal{L}_t$ that maximize the coverage $P(\mathcal{C})$, thereby improving the learner's generalization ability with limited labeling budget.

The purity of the neighborhood $B(x_i, r)$ measures the extent to which the samples within the neighborhood share the same class label as its center. Therefore, it reflects how reliably the chosen sample x_i represents a single class within its local neighborhood. Formally, the purity is defined through two definition below

Definition 4 (***Purity Ball***). *Given a point $x \in \mathcal{X}$ and its neighborhood $B(x, r)$, suppose that x has class label $f(x) = y$. The ball $B(x, r)$ is called* ***purity ball*** *if*

$$\forall x' \in \mathcal{X}, ||x' - x|| \leq r, f(x') = f(x). \tag{23}$$

Definition 5 (***Purity Region***). *Given a covered region $\mathcal{C}(\mathcal{L}_t, r) = \bigcup_{i=1}^{|\mathcal{L}_t|} B(x_i, r)$, the* ***purity*** *of this region, denoted by $\pi(\mathcal{C}(\mathcal{L}_t, r))$, is defined as follows:*

$$\pi(\mathcal{C}(\mathcal{L}_t, r)) = P(\{x : B(x, r) \textit{ is a purity ball } | x \in \mathcal{L}_t\}). \tag{24}$$

ProbCover (Yehuda et al., 2022) further establishes a theorem that expresses an upper bound on the generalization error minimization objective in Eq. 2 in terms of the coverage and purity of the selected region, as follows:

$$\mathbb{E}\big[\hat{f}_{\hat{\theta}}(x) \neq f(x)\big]_{\mathcal{X}} \leq (1 - P(\mathcal{C}(\mathcal{L}_t, r))) + (1 - \pi(\mathcal{C}(\mathcal{X}, r))) \tag{25}$$

Eq. 25 indicates that minimizing the generalization objective cannot be achieved by maximizing coverage alone; it also requires maximizing the purity of the total space. It is worth noting that the upper bound in Eq. (25) can be highly sensitive to the radius selected in ProbCover. To mitigate this issue, MaxHerding (Bae et al., 2024) and UHerding (Bae et al., 2025) incorporate kernel herding (Chen et al., 2010) into the coverage-purity framework, leading to a more general and stable upper bound for the objective in Eq. (25):

$$\mathbb{E}\big[\hat{f}_{\hat{\theta}}(x) \neq f(x)\big]_{\mathcal{X}} \leq \left(1 - \mathbb{E}_{\mathbf{x}}\left[\max_{\mathbf{x}' \in \mathcal{L}} \sigma(\mathbf{x}, \mathbf{x}')\right]\right) + \mathbb{E}_{\mathbf{x}}\left[\max_{\mathbf{x}' : f(\mathbf{x}) \neq f(\mathbf{x}')} \sigma(\mathbf{x}, \mathbf{x}')\right] \tag{26}$$

The first term in Eq. (26) refers to the coverage, where $P(\mathcal{C}_\sigma) = \mathbb{E}_{\mathbf{x}}\left[\max_{\mathbf{x}' \in \mathcal{L}} \sigma(\mathbf{x}, \mathbf{x}')\right]$, while the second term quantifies impurity, defined as $Imp(\mathcal{C}_\sigma) = \mathbb{E}_{\mathbf{x}}\left[\max_{\mathbf{x}' : f(\mathbf{x}) \neq f(\mathbf{x}')} \sigma(\mathbf{x}, \mathbf{x}')\right]$. In practice, the corresponding coverage and impurity terms can be estimated using Monte Carlo approximation as follows:

$$P(\mathcal{C}_\sigma) = \frac{1}{|\mathcal{U}_t|} \sum_{\mathbf{x}_n} \max_{\mathbf{x}' \in \mathcal{L}_t} \sigma\left(\mathbf{x}_n, \mathbf{x}'; g\right) \text{ with } \mathbf{x}_n \in \mathcal{U}_t \tag{27}$$

$$Imp(\mathcal{C}_\sigma) = \frac{1}{|\mathcal{U}_t|} \sum_{\mathbf{x}_n} \max_{\mathbf{x}' \in \mathcal{L}_t | f(\mathbf{x}') \neq f(\mathbf{x}_n)} \sigma\left(\mathbf{x}_n, \mathbf{x}'; g\right) \text{ with } \mathbf{x}_n \in \mathcal{U}_t \tag{28}$$

where g is the feature extractor, and σ is a non-negative kernel function. Both MaxHerding and UHerding apply the Gaussian kernel $\sigma\left(\mathbf{x}_n, \mathbf{x}'; g\right) = \exp\left(-\frac{1}{2\epsilon^2}||g(\mathbf{x}') - g(\mathbf{x}_n)||^2\right)$, in their formulations. Following these works, we also employs the Gaussian kernel throughout this study.

We use Eqs. (27) and (28) to analyze the performance of AL methods in Appendix F. We also emphasize that, although computing the purity or impurity requires the ground-truth labels of all samples in that set, these labels are used exclusively for post-hoc evaluation. They are never accessed during the active learning selection process and therefore have no influence on the sampling decisions made by any of the evaluated AL methods.

D PROOFS OF THEOREM 1

We now present the complete formulation of **Theorem 1**, followed by its detailed proof.

Theorem 1 (Finite Convergence of Self-Correction)**.** *Consider a search region $(\beta_i)^{(0)}$ associated with centroid c_i and class label $y_i = f(c_i)$. The search region is bounded by $M-1$ hyperplanes $\{(\alpha_{i,j})\}_{i\neq j}$ defined in Eq. (3) and Eq. (4). Without loss of generality, we assume that the centroid c_i lies on the negative side of every hyperplane. Then the region $(\beta_i)^{(s)}$ can be represented as:*

$$(\beta_i)^{(s)} = \bigcap_{j\neq i} \left\{ x \in \mathbb{R}^d : (\overrightarrow{n_{i,j}})^T x < \tau_{i,j}^{(s)} \right\}, \tag{29}$$

where $\tau_{i,j}^{(s)} = -b_{i,j}^{(s)}$ and s denotes the self-correction step. Assume that there exists a non-empty fixed ***pure region*** *$(\beta_i)^* \subseteq (\beta_i)^{(0)}$ of the form:*

$$(\beta_i)^* = \bigcap_{j\neq i} \left\{ x \in \mathbb{R}^d : (\overrightarrow{n_{i,j}})^T x < \tau_{i,j}^{*} \right\}, \tag{30}$$

such that $f(x) = y_i \; \forall x \in (\beta_i)^*$, $c_i \in (\beta_i)^*$ and $\tau_{i,j}^* \leq \tau_{i,j}^{(0)} \; \forall j \neq i$. At every correction step s, suppose that an incorrectly assigned sample $x^{(s)} \in (\beta_i)^{(s)}$ with $f(x^{(s)}) \neq y_i$ is detected. The algorithm selects a boundary (α_{i,j^*}) for which:

$$(\overrightarrow{n_{i,j^*}})^T x^{(s)} \geq \tau_{i,j^*}^*, \tag{31}$$

and updates only its threshold according to

$$\tau_{i,j^*}^{(s+1)} \in \left[\tau_{i,j^*}^*, (\overrightarrow{n_{i,j^*}})^T x^{(s)} \right), \tag{32}$$

while all other thresholds remain unchanged. Then the self-correction procedure satisfies the following properties:

1. *The sequence of search regions is monotonically shrinking: $(\beta_i)^{(s+1)} \subseteq (\beta_i)^{(s)}$.*
2. *The pure region is preserved throughout the correction process: $(\beta_i)^* \subseteq (\beta_i)^{(s)}$.*
3. *The centroid is preserved: $c_i \in (\beta_i)^{(s)}$ in every self-correction step s.*
4. *Every wrong sample is permanently excluded from the search region if it satisfies the update condition in Eq. (11).*
5. *If the active learning pool is finite, the self-correction algorithm terminates after a finite number of updates.*

Consequently, after a finite number of correction steps, the bias vector b_i stops changing and the resulting search region $(\beta_i)^{(S)} = (\beta_i)^* \cup \mathcal{N}_i^{(S)}$ where $\mathcal{N}_i^{(S)}$ is the set of incorrectly assigned samples that violate the condition in Eq. (11). The union of all resulting search regions $\Omega_{safe} = \bigcup_i (\beta_i)^{(S)}$ is a complete ***safe zone****.*

Proof. We prove the theorem through five steps.

Step 1: Each correction strictly shrinks the search region. At correction step s, the incorrectly assigned point $x^{(s)}$ belongs to $\beta_i^{(s)}$. Hence,

$$(\overrightarrow{n_{i,j^*}})^T x^{(s)} \leq \tau_{i,j^*}^{(s)}. \tag{33}$$

According to Eq. (32),

$$\tau_{i,j^*}^{(s+1)} < (\overrightarrow{n_{i,j^*}})^T x^{(s)} \leq \tau_{i,j^*}^{(s)}. \tag{34}$$

Therefore,

$$\tau_{i,j^*}^{(s+1)} < \tau_{i,j^*}^{(s)}. \tag{35}$$

All other thresholds remain unchanged. Thus, the corresponding half-space becomes strictly smaller, implying

$$(\beta_i)^{(s+1)} \subseteq (\beta_i)^{(s)}. \tag{36}$$

Step 2: The pure region is never removed. From Eq. (32), every updated threshold satisfies:

$$\tau_{i,j}^{(s)} \geq \tau_{i,j}^{*}. \tag{37}$$

For any $x \in (\beta_i)^*$, by the definition in Eq. (30),

$$(\overrightarrow{n_{i,j}})^T x < \tau_{i,j}^{*} \leq \tau_{i,j}^{(s)},\ \forall j \neq i. \tag{38}$$

Therefore,

$$x \in (\beta_i)^{(s)}, \tag{39}$$

which proves

$$(\beta_i)^* \subseteq \beta_i^{(s)}\ \forall s. \tag{40}$$

Hence, the self-correction procedure cannot shrink the search region beyond the assumed pure region.

Step 3: The centroid is preserved. This result can be inferred directly from Step 2. Because we assume $c_i \in (\beta_i)^*$, we have:

$$c_i \in \beta_i^{(s)}\ \forall s \tag{41}$$

Step 4: The wrong sample is excluded permanently if it satisfies the update condition in Eq. (11). If an incorrectly assigned sample $x^{(s)}$ satisfies the update condition in Eq.(11), the self-correction step is activated. From Eq. (32) we have:

$$(\overrightarrow{n_{i,j^*}})^T x^{(s)} > \tau_{i,j^*}^{(s+1)}. \tag{42}$$

Hence, $x^{(s)}$ violates the updated half-space constraint and therefore

$$x^{(s)} \notin (\beta_i)^{(s+1)}. \tag{43}$$

Since the thresholds are monotonically non-increasing in subsequent iterations, i.e.,

$$\tau_{i,j}^{(s+2)} \leq \tau_{i,j}^{(s+1)}, \tag{44}$$

the excluded sample can never re-enter the region. Therefore, every correction permanently removes at least one incorrectly assigned sample.

Step 5: Finite convergence. Let

$$\mathcal{N}_i = \left\{ x \in \mathcal{U}_t \cap (\beta_i)^{(0)} : f(x) \neq y_i \right\} \tag{45}$$

denote the set of incorrectly assigned samples in the initial search region. Let $\widehat{\mathcal{N}}_i \subset \mathcal{N}_i$ be the set of incorrectly assigned samples that satisfy the update condition in Eq. (11). Since pool-based active learning operates on a finite unlabeled pool $\mathcal{U}_t$, we have

$$|\mathcal{N}_i| < \infty. \tag{46}$$

From Step 5, every self-correction update permanently eliminates at least one element of $\mathcal{N}_i$ if it satisfies the update condition in Eq. (11). Because previously removed samples cannot re-enter the search region, the number of incorrectly assigned samples remaining after each correction monotonically decreases:

$$|\mathcal{N}_i^{(s+1)}| \leq |\mathcal{N}_i^{(s)}|. \tag{47}$$

Since this quantity is a non-negative integer, it can decrease only a finite number of times. Therefore, after at most $|\widehat{\mathcal{N}}_i|$ correction steps, we obtain a set of incorrectly assigned samples that violate the condition in Eq. (11) $\mathcal{N}_i^{(S)}$, and $|\mathcal{N}_i^{(S)}| \geq 0$, for some finite $S \leq |\widehat{\mathcal{N}}_i|$. At this point, no remaining samples can trigger further self-correction updates. Hence,

$$b_i^{(S+1)} = b_i^{(S)}, \tag{48}$$

and the self-correction algorithm converges. Combining Eq. (36) and Eq. (41), the final region satisfies

$$(\beta_i)^* \subseteq (\beta_i)^{(S)} \subseteq (\beta_i)^{(0)}. \tag{49}$$

This completes the proof. □

E PROOFS OF PROPOSITIONS 1, 2 AND 3

Proposition 1. *Given the unlabeled pool $\mathcal{U}_t$, and a set of centroids $\{c_i\}_{i=1}^M$, any sample $x \in \mathcal{U}_t$ belongs to the search region (β_i) if and only if c_i is the nearest centroid to x.*

$$x \in (\beta_i) \iff i = \arg\min_{j \in \{1,\dots,M\}} \|x - c_j\|.$$

Proof. We first prove the forward direction: if $x \in (\beta_i)$, then c_i is the nearest centroid to x. Suppose, for contradiction, that there exists another centroid c_j, with $j \neq i$, that is closer to x than c_i. Then,

$$d(c_j, x) = \|c_j - x\| < \|c_i - x\| = d(c_i, x).$$

For the separating hyperplane $(\alpha_{i,j})$, we have the signed Euclidean distance of a point z to the hyperplane as

$$d_\alpha(z) = (\overrightarrow{n_{i,j}})^T \odot (z - a_{i,j}),$$

where $\overrightarrow{n_{i,j}}$ is the normal vector of $(\alpha_{i,j})$ and $a_{i,j}$ is the midpoint between c_i and c_j. Without loss of generality, we choose the orientation of $\overrightarrow{n_{i,j}}$ such that

$$d_\alpha(c_i) < 0 < d_\alpha(c_j).$$

Since $\alpha_{i,j}$ passes through the midpoint $a_{i,j}$, the two centroids are symmetric with respect to the hyperplane. Therefore,

$$d_\alpha(c_i) + d_\alpha(c_j) = 0.$$

Moreover, since $x \in (\beta_i)$, the definition of the search region in Eq. (5) implies that x and c_i lie on the same side of every separating hyperplane associated with (β_i). In particular, we have:

$$d_\alpha(c_i) d_\alpha(x) > 0. \tag{50}$$

Let γ denote the line passing through c_i and c_j, $\bar{x}$ is the orthogonal projection of x onto γ, and ω is the line going through x and $\bar{x}$. Since ω and $(\alpha_{i,j})$ are perpendicular to γ, ω and $(\alpha_{i,j})$ are parallel. Therefore, $\bar{x} \in (\beta_i)$ and $d_\alpha(\bar{x}) = d_\alpha(x)$. There are two possible configurations: either $\bar{x}$ lies between c_i and $a_{i,j}$, or c_i lies between $\bar{x}$ and c_j. In both cases, the squared distances from $\bar{x}$ to the two centroids c_i and c_j can be expressed as follows:

$$d^2(c_i, \bar{x}) = (d_\alpha(c_i) - d_\alpha(\bar{x}))^2 = (d_\alpha(c_i) - d_\alpha(x))^2,$$
$$d^2(c_j, \bar{x}) = (d_\alpha(c_j) - d_\alpha(\bar{x}))^2 = (d_\alpha(c_j) - d_\alpha(x))^2.$$

Using the Pythagorean theorem on the triangle formed by $\Delta(c_i, x, \bar{x})$, and $\Delta(c_j, x, \bar{x})$, we have:

$$d^2(c_i, x) = d^2(c_i, \bar{x}) + d^2(\bar{x}, x) = (d_\alpha(c_i) - d_\alpha(x))^2 + d^2(\bar{x}, x),$$
$$d^2(c_j, x) = d^2(c_j, \bar{x}) + d^2(\bar{x}, x) = (d_\alpha(c_j) - d_\alpha(x))^2 + d^2(\bar{x}, x).$$

From the assumption $d(c_j, x) < d(c_i, x)$, it follows that:

$$\begin{aligned}
&(d_\alpha(c_j) - d_\alpha(x))^2 < (d_\alpha(c_i) - d_\alpha(x))^2 \\
\Leftrightarrow\ &(d_\alpha(c_j) - d_\alpha(x))^2 - (d_\alpha(c_i) - d_\alpha(x))^2 < 0 \\
\Leftrightarrow\ &(d_\alpha(c_j) - d_\alpha(c_i))(d_\alpha(c_j) + d_\alpha(c_i) - 2d_\alpha(x)) < 0 \\
\Leftrightarrow\ &(-2d_\alpha(c_i))(-2d_\alpha(x)) < 0 \\
\Leftrightarrow\ &d_\alpha(c_i) d_\alpha(x) < 0
\end{aligned}$$

which contradicts the assumption $x \in (\beta_i)$ in Eq. (50). Therefore, no centroid c_j, $j \neq i$, can be closer to x than c_i. Hence, c_i is a nearest centroid to x.

We now prove the backward direction: if c_i is the nearest centroid to x, then $x \in (\beta_i)$. We again proceed by contradiction. Suppose that c_i is the nearest centroid to x, but $x \notin (\beta_i)$. By the definition of the search regions, there must exist some $j \neq i$ such that x lies on the same side of the separating hyperplane $\alpha_{i,j}$ as c_j. Therefore,

$$d_\alpha(c_j) d_\alpha(x) > 0 \tag{51}$$

Following the same geometric construction used in the forward direction, let γ be the line passing through c_i and c_j, and let $\bar{x}$ denote the orthogonal projection of x onto γ. The squared distances from $\bar{x}$ to the two centroids can then be expressed as:

$$d^2(c_i, \bar{x}) = (d_\alpha(c_i) - d_\alpha(\bar{x}))^2 = (d_\alpha(c_i) - d_\alpha(x))^2,$$
$$d^2(c_j, \bar{x}) = (d_\alpha(c_j) - d_\alpha(\bar{x}))^2 = (d_\alpha(c_j) - d_\alpha(x))^2.$$

Since c_i is assumed to be the nearest centroid to x, we have:

$$d(c_i, x) < d(c_j, x). \tag{52}$$

Applying the Pythagorean theorem as in the forward direction and canceling the common term $d^2(\bar{x}, x)$ gives:

$$\begin{aligned} &(d_\alpha(c_i) - d_\alpha(x))^2 < (d_\alpha(c_j) - d_\alpha(x))^2 \\ \Leftrightarrow\ &(d_\alpha(c_i) - d_\alpha(x))^2 - (d_\alpha(c_j) - d_\alpha(x))^2 < 0 \\ \Leftrightarrow\ &(d_\alpha(c_i) - d_\alpha(c_j))(d_\alpha(c_j) + d_\alpha(c_i) - 2d_\alpha(x)) < 0 \\ \Leftrightarrow\ &(-2d_\alpha(c_j))(-2d_\alpha(x)) < 0 \\ \Leftrightarrow\ &d_\alpha(c_j) d_\alpha(x) < 0 \end{aligned}$$

which contradicts Eq. (51). Hence, our assumption that $x \notin (\beta_i)$ is false, and we conclude that $x \in (\beta_i)$. This completes the proof of the backward direction. □

Proposition 2. *For any subset $\mathcal{Z} \subseteq \mathcal{X}$, the empirical safe coverage is upper bounded by the corresponding empirical uncertainty coverage:*

$$\widehat{SC}(\mathcal{Z}) \leq \widehat{UC}_{\sigma_\gamma}(\mathcal{Z}). \tag{53}$$

Proof. From Eq. (13), for any pair $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, we have

$$0 \leq \tilde{\sigma}_\gamma(\mathbf{x}, \mathbf{x}'; g) \leq \sigma_\gamma(\mathbf{x}, \mathbf{x}'; g). \tag{54}$$

Therefore, for every $\mathbf{x}_n$,

$$\max_{\mathbf{x}' \in \mathcal{Z}} \tilde{\sigma}_\gamma(\mathbf{x}_n, \mathbf{x}'; g) \leq \max_{\mathbf{x}' \in \mathcal{Z}} \sigma_\gamma(\mathbf{x}_n, \mathbf{x}'; g). \tag{55}$$

Since $U(\mathbf{x}_n; \hat{f}) \geq 0$, multiplying both sides by the uncertainty term and averaging over all N samples preserves the inequality. Hence,

$$\widehat{SC}(\mathcal{Z}) \leq \widehat{UC}_{\sigma_\gamma}(\mathcal{Z}), \tag{56}$$

which completes the proof. □

Proposition 3. *For any search region (β_i), the empirical classification error satisfies*

$$\varepsilon_i \leq (1 - \phi_i) + (1 - \pi_i).$$

In particular, if (β_i) is pure, i.e., $\pi_i = 1$, then $\varepsilon_i \leq 1 - \phi_i$ Therefore, $\phi_i \to 1 \Longrightarrow \varepsilon_i \to 0$ for a pure search region.

Proof. Let $y_i = f(c_i)$ denote the reference label of region (β_i). For any sample $x \in (\beta_i)$, if $\hat{f}(x) \neq f(x)$, then at least one of the following two events must occur:

1. $\hat{f}(x) \neq y_i$, i.e., the learner disagrees with the region label; or
2. $f(x) \neq y_i$, i.e., the true label of x disagrees with the region label.

Therefore, for every $x \in (\beta_i)$,

$$\mathbb{1}\left[\hat{f}(x) \neq f(x)\right] \leq \mathbb{1}\left[\hat{f}(x) \neq y_i\right] + \mathbb{1}[f(x) \neq y_i]\,. \tag{57}$$

Averaging both sides over all samples in (β_i) gives

$$\varepsilon_i \leq \frac{1}{|(\beta_i)|} \sum_{x \in (\beta_i)} \mathbb{1}\left[\hat{f}(x) \neq y_i\right] + \frac{1}{|(\beta_i)|} \sum_{x \in (\beta_i)} \mathbb{1}[f(x) \neq y_i] \tag{58}$$

$$= (1 - \phi_i) + (1 - \pi_i). \tag{59}$$

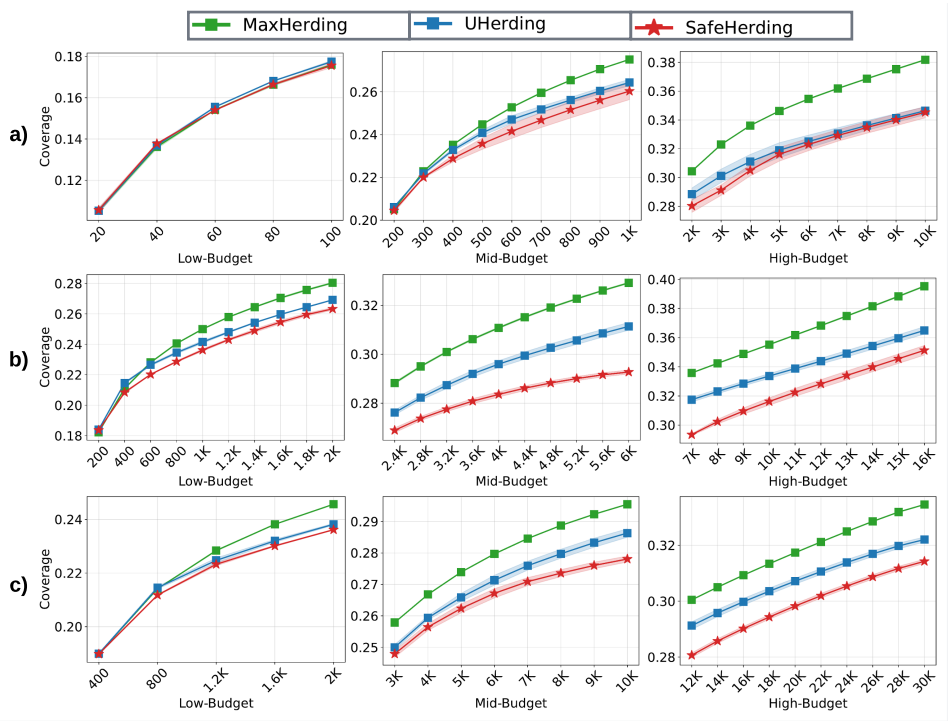

Figure 4: Coverage comparison across different datasets: (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

When (β_i) is pure, we have $\pi_i = 1$, and therefore

$$\varepsilon_i \leq 1 - \phi_i. \tag{60}$$

Hence, as $\phi_i \to 1$, the upper bound approaches zero, which implies

$$\varepsilon_i \to 0. \tag{61}$$

This completes the proof. □

F More Ablation Studies

F.1 Coverage and Purity Comparison

To further investigate the performance gains of SafeHerding, we compare the coverage and impurity of MaxHerding, UHerding, and SafeHerding, using the definitions introduced in Appendix C. As shown in Figure 4, SafeHerding consistently attains the lowest coverage, while MaxHerding achieves the highest coverage across all experimental settings. This behavior is consistent with their respective formulations: the uncertainty coverage adopted by UHerding is contained within the generalized coverage of MaxHerding, whereas **Proposition 2** establishes the uncertainty coverage as an upper bound on the safe coverage of SafeHerding. As the annotation budget grows, the three methods select more samples and therefore expand their coverage over the data space. Nevertheless, this increase in coverage is accompanied by higher impurity, which may negatively affect learner performance. Figure 5, reveals an almost opposite trend for impurity: MaxHerding generally produces the highest impurity, UHerding ranks in between, and SafeHerding achieves the lowest impurity in most cases. Together with the experimental results presented in Section 5, these observations suggest that maximizing coverage alone may be suboptimal when it substantially sacrifices purity. Despite covering the largest portion of the data space, MaxHerding performs worse than both UHerding and SafeHerding. By contrast, SafeHerding has the strongest predictive performance while maintaining considerably lower impurity, underscoring the importance of preserving purity during coverage expansion.

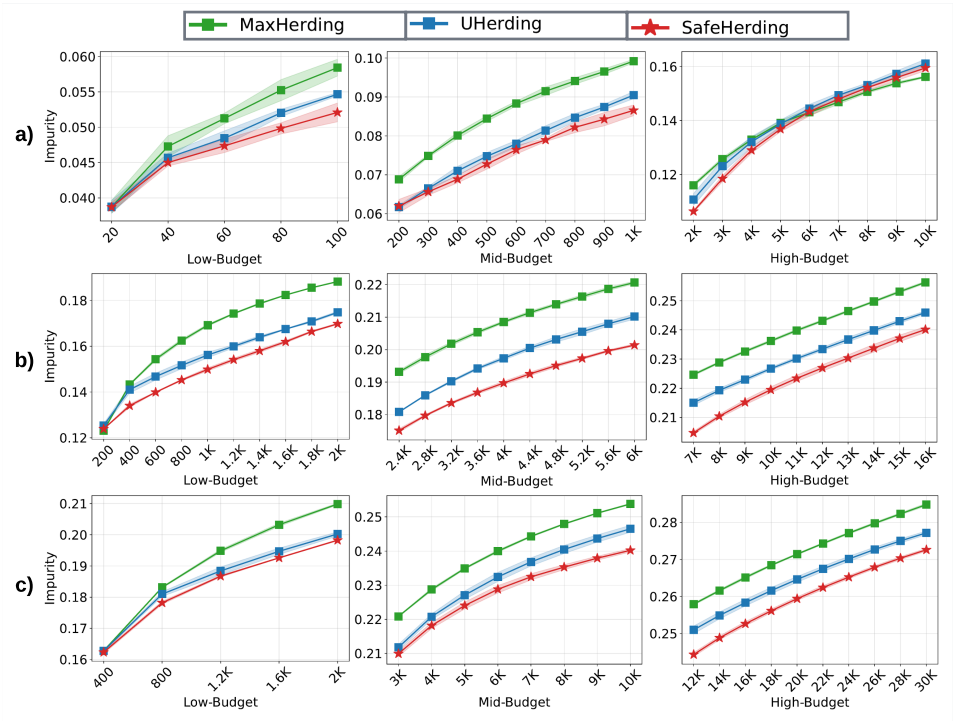

Figure 5: Impurity comparison across different datasets: (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

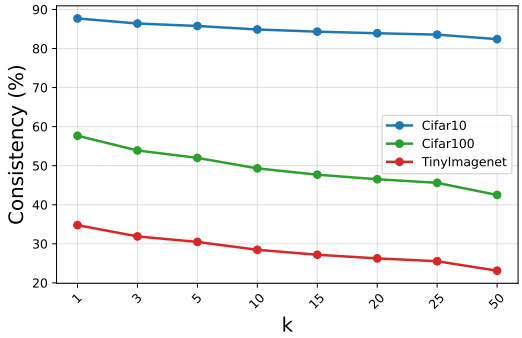

Figure 6: The label consistency (%) of CIFAR10, CIFAR100, and TinyImageNet.

F.2 Understanding the Sampling Procedure in the Coverage–Purity Framework

Following the analysis in Appendix F.1, we conduct a more detailed examination of the sampling behavior of MaxHerding, UHerding, and SafeHerding to clarify their differences in coverage and purity. As a preliminary step, we characterize the local structure of each dataset through a k-nearest-neighbor (k-NN) label-consistency analysis, with the results shown in Figure 6. For each sample x, we retrieve its k nearest neighbors and calculate the fraction of those neighbors that have the same ground-truth label as x. We then average this fraction over all samples in the dataset. By varying k from 1 to 50, we assess how label consistency evolves as the local neighborhood becomes larger.

The resulting curves offer an empirical view of the local class structure in the feature space. CIFAR10 maintains the highest label consistency across neighborhood sizes, indicating a relatively simple and locally coherent geometry. Since CIFAR10 contains only 10 classes and many samples per class, samples from the same category are more likely to form compact and label-consistent neighborhoods under an informative feature representation. By comparison, CIFAR100 and TinyImageNet contain

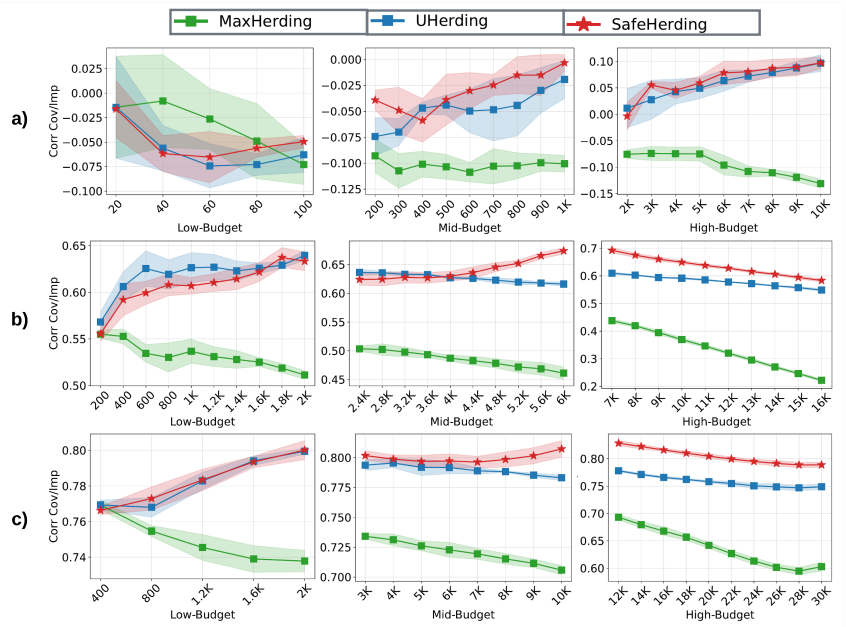

Figure 7: The Spearman correlation between the coverage and impurity in (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

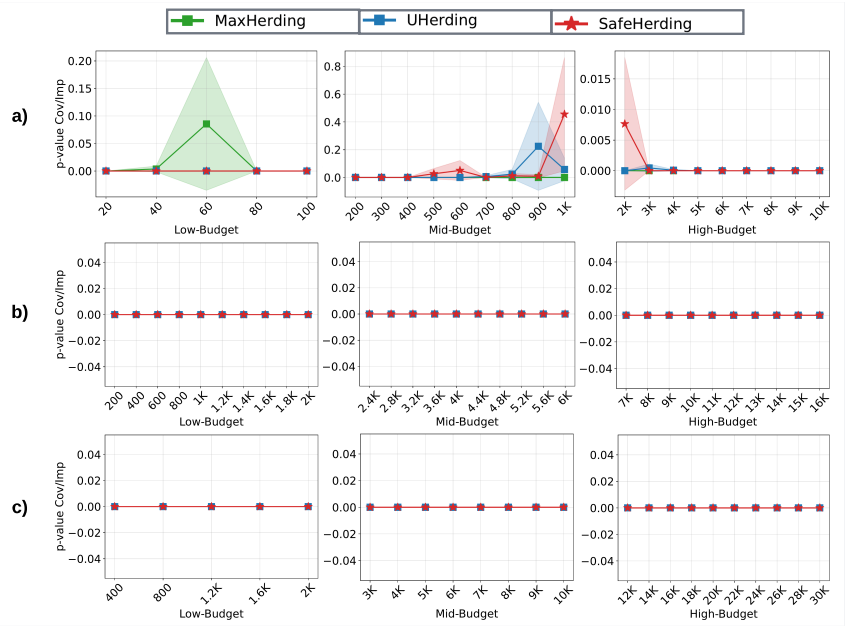

Figure 8: The p-value between the coverage and impurity in (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

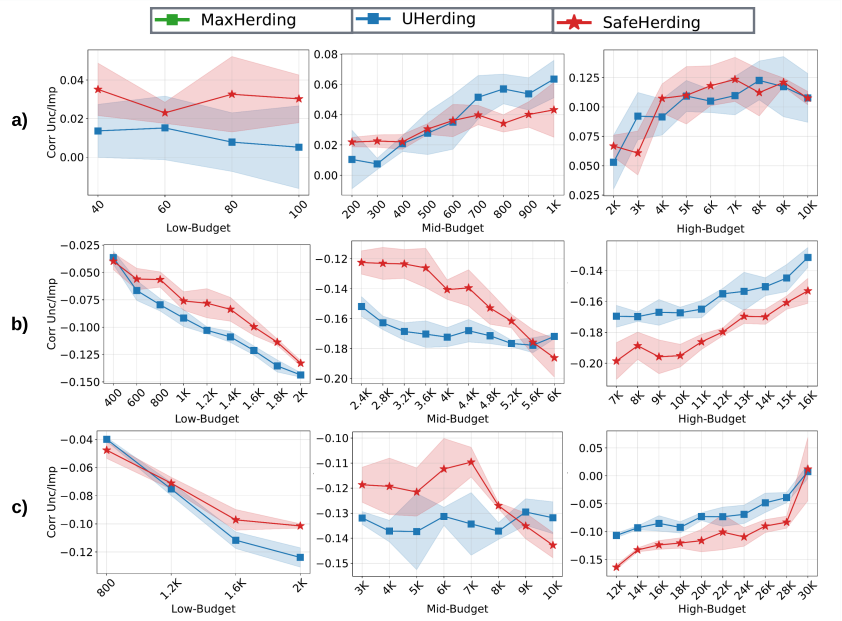

Figure 9: The Spearman correlation between the uncertainty and impurity in (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

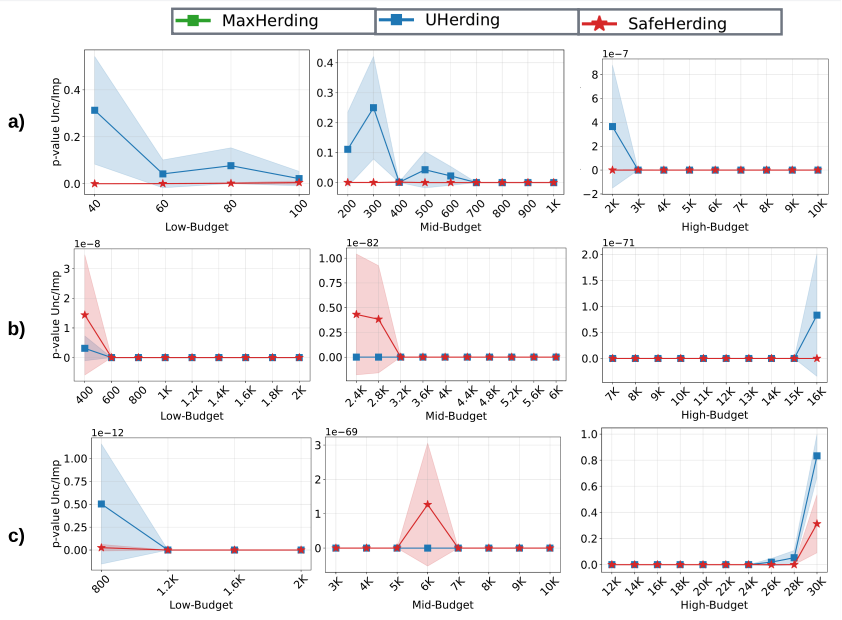

Figure 10: The p-value between the uncertainty and impurity in (a) CIFAR10, (b) CIFAR100, and (c) TinyImageNet.

Method	Low					Middle			High				
	C10	C100(s)	Tiny.	C100(t)	Dom.	C10	C100(s)	Tiny.	C10	C100(s)	Tiny.	C100(t)	Dom.
Entropy	-4.4	-16.1	-10.8	–	–	-4.0	-6.2	-13.7	**+4.1**	+0.9	+1.2	–	–
Margin	-2.7	-3.2	+0.6	+7.5	+1.3	-3.5	-0.3	+0.6	+3.5	+3.5	**+4.3**	+3.3	+2.1
BADGE	-2.4	+0.2	+5.8	+4.1	+1.4	-1.0	+1.7	+2.8	+3.9	+2.4	+1.5	+2.8	+2.3
ALFA-M	-11.0	-0.9	+6.9	–	–	-5.2	-1.3	+7.1	-4.2	-1.7	+1.0	–	–
Weighted k-Means	+3.3	+7.0	+1.5	-1.6	-0.8	+0.5	+1.1	-0.7	+0.5	+0.0	+2.7	+1.8	-0.3
Coreset	-14.9	-34.3	-33.7	–	–	-34.5	-34.4	-34.7	-3.5	-14.3	-11.4	–	–
ActiveFT	–	–	–	+3.6	+0.0	–	–	–	–	–	–	-0.5	+0.0
Typiclust	+11.8	+21.7	-0.5	+10.2	+0.5	+7.3	+6.6	-2.4	-1.6	-0.3	-1.2	-0.2	+0.4
MaxHerding	+22.9	+18.5	+32.6	+10.1	+0.9	+10.1	+3.8	+4.6	-0.5	-5.9	-6.5	+2.7	+0.7
UHerding	+24.2	+31.6	+67.7	+9.1	+2.1	+11.9	+10.9	+9.5	+3.2	+3.6	-0.8	**+4.3**	+2.6
SafeHerding	**+30.5**	**+35.5**	**+72.7**	**+10.8**	**+2.3**	**+14.1**	**+12.2**	**+14.1**	+4.0	**+4.9**	+3.4	+3.1	**+2.7**

Table 2: Comparison of the gain in mean accuracy, measured by ΔAcc (%), relative to Random sampling across different datasets and annotation-budget regimes. For each setting, the best result is highlighted in **bold**, while the second-best result is underlined. For CIFAR100, (s) denotes the supervised learning setting, whereas (t) refers to the transfer learning setting. The notation "–" indicates that the method was not implemented in this setting.

100 and 200 classes, respectively, which introduces greater inter-class diversity and more complex local arrangements. As the neighborhood size increases, samples from different classes are therefore more likely to be included, causing label consistency to decline more rapidly than on CIFAR10. This finding indicates that constructing and preserving high-purity regions is intrinsically more difficult on CIFAR100 and TinyImageNet.

Considering the distinct geometric properties of the three datasets, we analyze the sampling characteristics of MaxHerding, UHerding, and SafeHerding through the interactions between coverage, purity and uncertainty. Figures 7 and 8 demonstrate the Spearman correlation coefficients together with their corresponding p-values for coverage and impurity. In most settings, the p-values are close to zero and correlation are higher than 0.5, suggesting a statistically significant and relatively strong association between the two quantities. On CIFAR10, the correlation is mainly negative and small, which is consistent with its relatively simple and locally coherent feature-space structure. In this setting, increasing coverage can still preserve largely homogeneous neighborhoods, enabling MaxHerding to achieve high purity despite directly optimizing for coverage. In contradiction, CIFAR100 and TinyImageNet exhibit strong positive correlations between coverage and impurity across all three methods, indicating that expanding coverage is generally associated with greater impurity. This trend provides an explanation for the higher impurity of MaxHerding on these more complex datasets, since its sampling objective places greater emphasis on coverage, as discussed in Appendix F.1.

We further investigate the relationship between uncertainty and impurity to explain the lower impurity achieved by UHerding and SafeHerding, with the results reported in Figures 9 and 10. The consistently small p-values in Figure 10 indicate that the connection between uncertainty and impurity is expected in most cases. In particular, on CIFAR100 and TinyImageNet, the two quantities are negatively correlated, suggesting that selecting more uncertain samples is associated with lower impurity in these settings. This behavior benefits UHerding and SafeHerding, both of which incorporate uncertainty information, whereas MaxHerding does not. However, uncertainty alone cannot fully explain the observed purity improvements, as the absolute Spearman correlation coefficients in Figure 9 remain relatively small. SafeHerding achieves lower impurity than UHerding in most cases, highlighting the contribution of the safe zone framework introduced in Section 3. By restricting selection to reliable and high-purity regions, SafeHerding more effectively controls impurity while still benefiting from uncertainty-based exploration.

F.3 Comparison to Random Sampling

Table 2 compares the accuracy gains of different active learning methods relative to Random sampling across datasets and annotation-budget regimes. In particular, the relative accuracy gains is computed as:

$$\Delta\mathrm{Acc}(\%) = \frac{\mathrm{MeanAcc}_{\mathrm{Method}} - \mathrm{MeanAcc}_{\mathrm{Random}}}{\mathrm{MeanAcc}_{\mathrm{Random}}} \times 100.$$

Method	Runtime (in seconds)					Memory (in GiB)				
	C10	C100(s)	Tiny.	C100(t)	Dom.	C10	C100(s)	Tiny.	C100(t)	Dom.
MaxHerding	46.39	39.33	101.3	100.8	138.5	11.18	8.45	9.69	9.02	10.80
UHerding	259.3	444.6	1095.6	980.4	902.8	11.25	8.46	9.71	9.11	11.02
SafeHerding	267.2	459.9	1118.3	1003.7	1736.6	11.28	8.86	10.06	10.12	12.36

Table 3: Comparison of the mean computational complexity of MaxHerding, UHerding, and SafeHerding in terms of runtime and memory consumption per active learning round. For CIFAR100, (s) denotes the supervised learning setting, whereas (t) refers to the transfer learning setting.

We report the performance improvement between each method and Random sampling at the final active learning round of each setting. While several baselines fall below Random sampling in some cases, SafeHerding consistently achieves positive improvements across all settings and ranks first in most of them. The largest gain is observed on TinyImageNet under the low-budget regime, where SafeHerding improves accuracy by 72.7% relative to Random sampling, exceeding the second-best method, UHerding, by more than 5%. Across most settings, SafeHerding also provides an additional improvement of approximately 1% to 5% over the strongest baseline. These results further demonstrate the effectiveness of the proposed safe zone framework and the resulting SafeHerding.

F.4 Complexity Comparison

For a better complexity comparison, we report the training time and memory of our proposed SafeHerding against two herding-based baselines, MaxHerding and UHerding, in Table 3. In terms of memory usage, all three methods incur comparable costs. Although SafeHerding generally requires slightly more memory, the additional overhead remains limited, reaching at most about 10% in our experiments, as observed on DomainNet. The dominant source of memory consumption for all three methods is the kernel herding procedure, which requires computing pairwise kernel values among samples and therefore storing a large kernel matrix. By contrast, the additional memory introduced by the safe zone framework is relatively small when the number of clusters is moderate, as on CIFAR10 and CIFAR100. This indicates that the proposed framework remains lightweight and memory-efficient.

Regarding runtime, MaxHerding is consistently the fastest method. UHerding incurs substantially higher computational cost because of its additional temperature estimation and uncertainty computation. SafeHerding exhibits a runtime comparable to UHerding in most settings, with the main exception of DomainNet. On this dataset, SafeHerding employs a relatively large number of clusters ($M = 690$), which increases the number of self-correction and agreement-skipping operations and consequently leads to higher runtime.

F.5 Effectiveness of Key Components

To evaluate the impact of each key component in our proposed SafeHerding, we perform an ablation study using two variants, defined as:

- **w/o skipping:** SafeHerding without the agreement-based skipping mechanism. Sample selection remains restricted to the refined search regions produced by the self-correction algorithm, without adaptively relaxing the safe-zone constraints.
- **w/o self-correction:** SafeHerding without the self-correction algorithm. Sample selection is performed within the initial search regions, whose union covers the entire data space, without further refinement to improve purity. The agreement-based skipping mechanism is still applied to these unrefined regions.

Figure 11 illustrates the impacts of key components on the capacity of SafeHerding in supervised learning setting. Figures 12 and 13 further examine their effects on coverage and impurity, respectively. Overall, removing either component consistently degrades performance across all evaluated settings, highlighting their importance to SafeHerding.

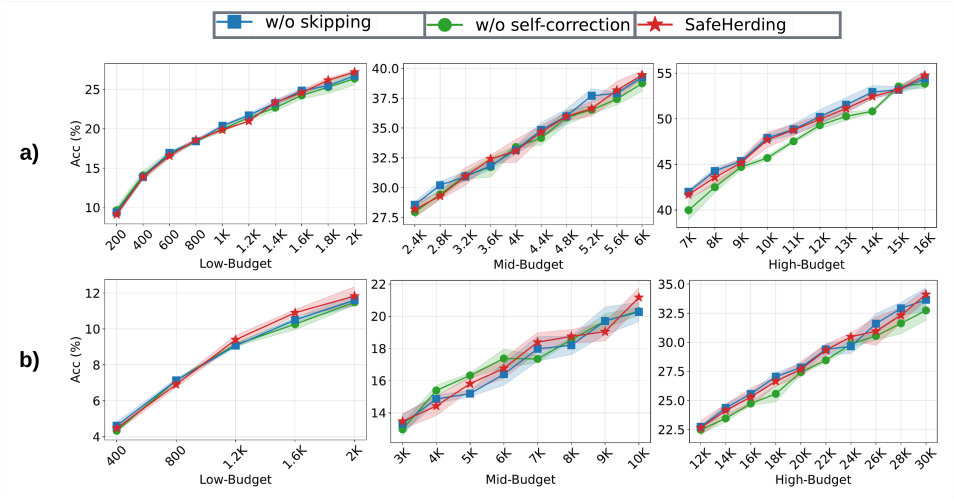

Figure 11: Impact of each component on supervised learning results, reported on (a) CIFAR100 and (b) TinyImageNet.

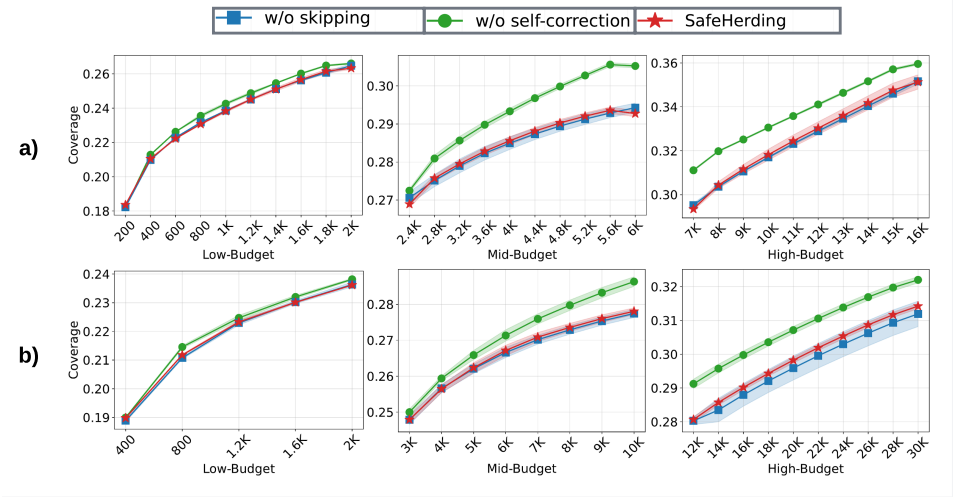

Figure 12: Impact of each component on the coverage: (a) CIFAR100 and (b) TinyImageNet.

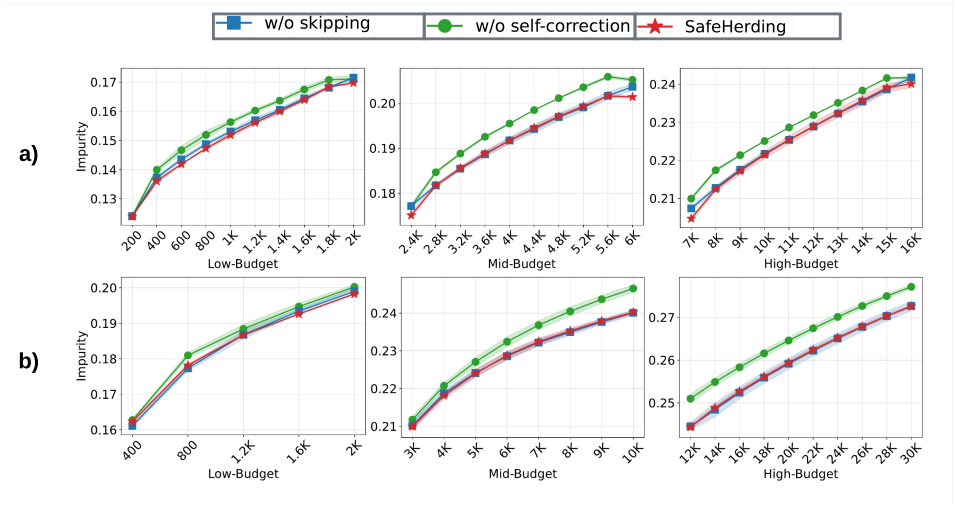

Figure 13: Impact of each component on the impurity: (a) CIFAR100 and (b) TinyImageNet.

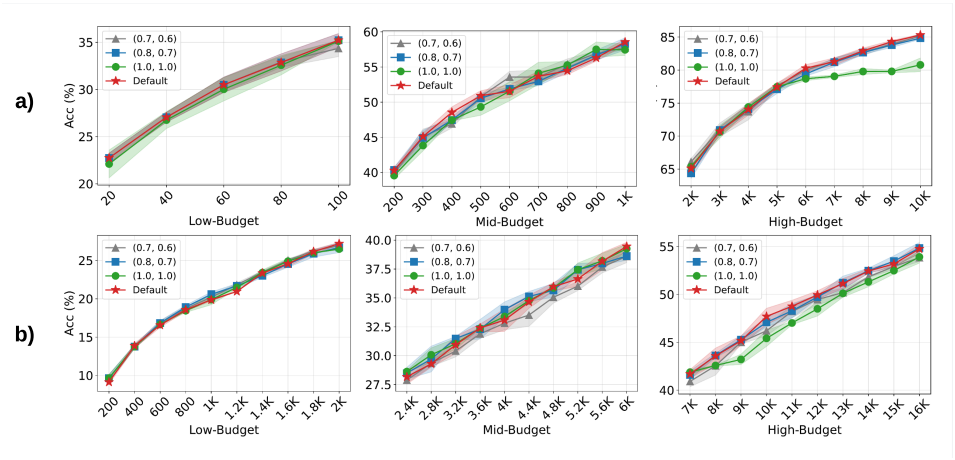

Figure 14: Supervised learning results with different thresholds on (a) CIFAR10 and (b) CIFAR100.

In particular, removing the self-correction algorithm (**w/o self-correction**) leads to a substantial performance decline, with accuracy reductions of 4.05% and 4.03% in the medium- and high-budget regimes on TinyImageNet, respectively. Without self-correction, sample selection is performed over the initial, unrefined search regions, resulting in broader coverage but increased impurity; consequently, lower classification accuracy.

In contrast, removing the agreement-based skipping mechanism (**w/o skipping**) yields impurity levels comparable to those of the complete SafeHerding method but slightly reduces coverage. This is because the sampler remains restricted to the refined safe regions, which may lead to redundant selections that contribute little additional information to the learner. These results highlight the complementary roles of the two components: self-correction preserves purity, whereas agreement-based skipping alleviates overly restrictive sampling and enables broader exploration. Combining both mechanisms is therefore important for maximizing SafeHerding's performance.

F.6 Sensitivity to Hyperparameters in SafeHerding

To assess the sensitivity of SafeHerding to the hyperparameters, we conduct ablation studies on the local and global agreement-based skipping thresholds, as well as the number of clusters. We adopt the supervised learning configuration in Section 5.1 as the default setting. Specifically, the local and

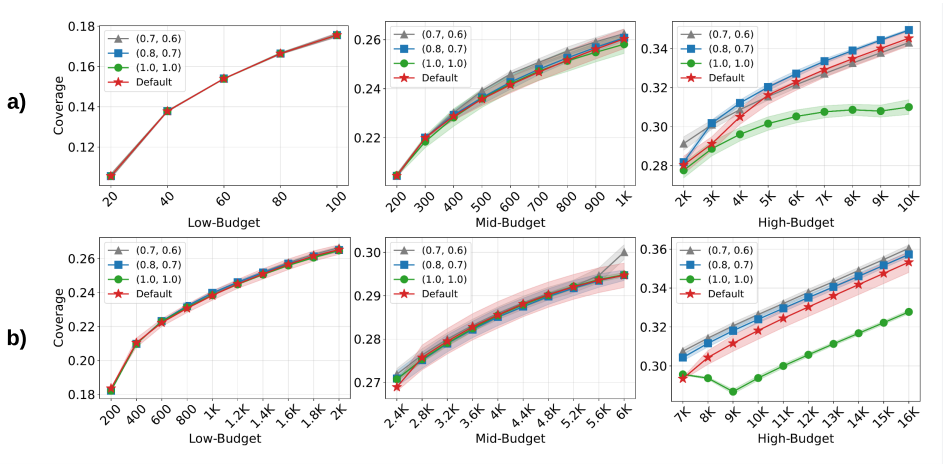

Figure 15: Coverage in different thresholds: (a) CIFAR10 and (b) CIFAR100.

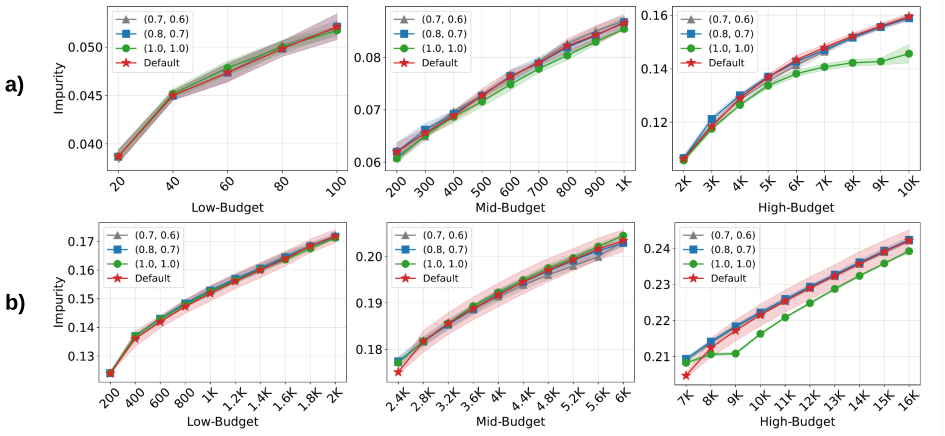

Figure 16: Impurity in different thresholds:(a) CIFAR10 and (b) CIFAR100.

global thresholds are set to $(0.9, 0.8)$, while the number of clusters is set to 200 for CIFAR-100 and 400 for Tiny ImageNet.

For the threshold ablation, we evaluate several pairs of local and global thresholds and report the results on CIFAR-10 and CIFAR-100 in Figure 14. Overall, the default setting consistently achieves the best performance. SafeHerding also remains competitive when the thresholds are set to $(0.7, 0.6)$ and $(0.8, 0.7)$. However, in the extreme case of $(1.0, 1.0)$, the performance degrades, particularly in the high-budget regime. Under this configuration, SafeHerding places excessive emphasis on purity. To further explain this behavior, we report the corresponding coverage and impurity in Figures15 and 16. A substantial gap can be observed between the extreme setting $(1.0, 1.0)$ and the other configurations in terms of both coverage and impurity. Nevertheless, its classification performance in Figure 14 remains inferior. This finding suggests that an appropriate balance between coverage and purity is required. Our default threshold setting provides such a balance by allowing SafeHerding to transition adaptively from purity-aware selection to broader coverage-aware exploration.

We further analyze the effect of the number of clusters in Figure 17. Increasing the number of clusters does not consistently improve SafeHerding and can even lead to lower performance in some cases. Moreover, a larger number of clusters increases the computational overhead of the self-correction mechanism. Therefore, we recommend the default cluster settings as they provide a favorable trade-off between computational efficiency and predictive performance.

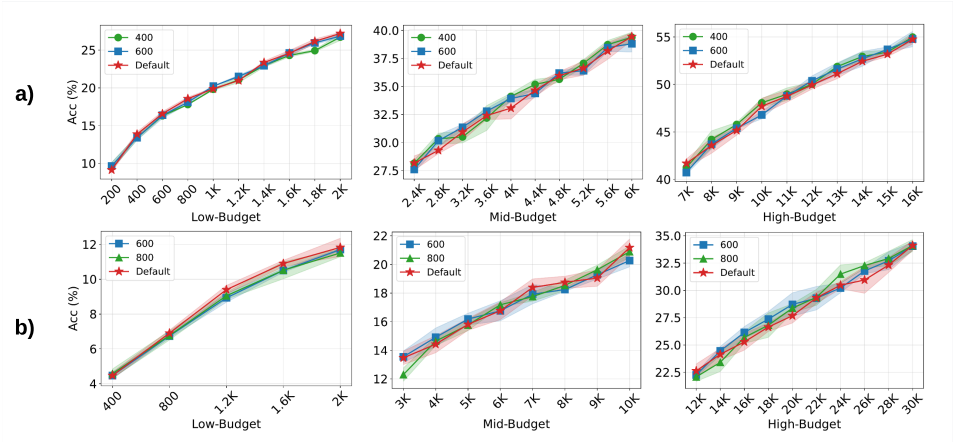

Figure 17: Supervised learning results with different numbers of clusters on (a) CIFAR100 and (b) TinyImageNet.

Table 4: Summary of the datasets used in our experiments.

Dataset	# Training	# Validation + Testing	Resolution	Classes
CIFAR10	50,000	10,000	32×32	10
CIFAR100	50,000	10,000	32×32	100
TinyImageNet	100,000	20,000	64×64	200
DomainNet-Real	120,906	52,041	224×224	345

G Experimental Details

G.1 Datasets

For the supervised learning setting, we evaluate the effectiveness of our proposed method on three widely used benchmarks: CIFAR10 (Krizhevsky & Hinton, 2009), CIFAR100 (Krizhevsky & Hinton, 2009), and TinyImageNet (Wu et al., 2017). For the transfer learning setting, we further evaluate its performance on CIFAR100 (Krizhevsky & Hinton, 2009) and DomainNet (Peng et al., 2019). The statistics of all datasets used in this study are presented in Table 4. Detailed descriptions of each dataset are provided as follows:

- CIFAR10 (Krizhevsky & Hinton, 2009): contains of $60,000$ color images in 10 classes. Each class has $6,000$ images 32×32. This dataset is divided into $50,000$ training images and $10,000$ testing images.
- CIFAR100 (Krizhevsky & Hinton, 2009): contains of $60,000$ color images in 100 fine-grained classes. Each class has 600 images 32×32, which contain 500 training images and 100 testing images.
- TinyImageNet (Wu et al., 2017): is a subset of ImageNet dataset, contains $120,000$ color images in 200 different classes. Each class has 500 training images, 50 validation images and 50 testing images 64×64.
- DomainNet (Peng et al., 2019): consists approximately $600,000$ images of 345 classes with 6 domains: Clipart, Real, Sketch, Infograph, Painting and Quickdraw. All images are resized to 224×224 before being fed into the network. Follow (Bae et al., 2025), we use the Real domain for our experiment.

G.2 Baselines

To evaluate the effectiveness of our proposed method, we compare it with the following baselines:

- Random: uniformly select random $\mathcal{B}$ samples at each active round.
- Margin (Scheffer et al., 2001): select $\mathcal{B}$ samples with the largest differences between the highest and second-highest predicted probabilities.
- Entropy (Wang & Shang, 2014): select $\mathcal{B}$ samples with the highest Shannon entropy.
- Weighted k-Means (Zhdanov, 2019): select $\mathcal{B}$ samples with sufficiently large margins that are closest to the centroids obtained from weighted k-means clustering.
- ALFA-Mix (Parvaneh et al., 2022): select $\mathcal{B}$ samples by identifying prediction inconsistencies induced by feature interpolation between labeled and unlabeled instances.
- Coreset (Sener & Savarese, 2018): select $\mathcal{B}$ samples which is expected to minimize the bound between an average loss over any given subset of the dataset and the remaining data points via the geometry of the data points.
- BADGE (Ash et al., 2020): select $\mathcal{B}$ diverse and uncertain samples using gradient embeddings to maximize label efficiency.
- Typiclust (Hacohen et al., 2022): select $\mathcal{B}$ samples with the highest "typicality" using m-Nearest Neighbors.
- ActiveFT (Xie et al., 2023): select $\mathcal{B}$ samples that represent the distribution of the entire unlabeled pool while maintaining diversity through optimization of a parametric model in the continuous space.
- MaxHerding (Bae et al., 2024): select $\mathcal{B}$ samples to maximize the generalized coverage.
- UHerding (Bae et al., 2025): select $\mathcal{B}$ samples to maximize the uncertainty-weighted coverage.

G.3 Implementation Details

We use a pretrained model to extract feature to compute coverage in our selection method, specifically backbone ResNet-18 (He et al., 2016) for supervised learning task and DeiT (Touvron et al., 2021) for transfer learning task. In supervised learning setting, we consider three annotation budgets as low-, mid-, and high- regimes for each dataset: $\mathcal{B} \in \{20, 100, 1000\}$ for CIFAR10, $\mathcal{B} \in \{200, 400, 1000\}$ for CIFAR100, and $\mathcal{B} \in \{400, 1000, 2000\}$ for TinyImageNet. For the task transfer experiments, we set $\mathcal{B} = 200$ and 800 for CIFAR100, and $\mathcal{B} = 700$ and 1400 for DomainNet, corresponding to low-budget and high-budget settings, respectively. For the skipping mechanism, we evaluate different local thresholds, $\lambda \in \{0.7, 0.8, 0.9, 1.0\}$, together with global thresholds, $\Lambda \in \{0.6, 0.7, 0.8, 1.0\}$. For the number of clusters, we try different values of M, as detailed in Section 5 and Appendix F. In all experiments, the model is re-initialized after each active learning round, either from random initialization or from pre-trained parameters, rather than being warm-started from the previous round. We implement our method SafeHerding by Pytorch. All the experiments are conducted on a GPU device NVIDIA GeForce RTX 4090 with 24GB memory.

H More Details About Our Work

H.1 Theorem 1 in Practice

In Section 3.2, we establish **Theorem 1** to guarantee the finite convergence of the self-correction algorithm. The theorem assumes that a fixed pure region $(\beta_i)^*$ is preserved throughout the refinement process, i.e., boundary updates do not exclude any portion of this region. However, in practice, translating the hyperplane boundaries may intersect and consequently shrink the underlying pure region.

To account for this possibility, let $(\beta_i)^{*,(s)} \subseteq (\beta_i)^{(s)}$ denote the pure region retained after the s-th self-correction step. Following the subsequent boundary update, we define the retained pure region as

$$(\beta_i)^{*,(s+1)} = (\beta_i)^{(s+1)} \cap (\beta_i)^{*,(s)}. \tag{62}$$

By construction, $(\beta_i)^{*,(s+1)} \subseteq (\beta_i)^{(s+1)}$. Moreover, since $(\beta_i)^{*,(s+1)} \subseteq (\beta_i)^{*,(s)}$, the resulting region preserves the purity of its predecessor. Therefore, provided that an initial pure region

$(\beta_i)^{*,(0)} \subseteq (\beta_i)^{(0)}$ exists and the centroid c_i is preserved in each boundary update, the refinement process retains a non-empty pure region at every iteration.

Under this condition, the thresholds $\tau^*_{i,j}$ in **Theorem 1** can be chosen with respect to the final retained pure region $(\beta_i)^{*,(S)}$, provided that this region is known or its preservation can be guaranteed. Since the search regions shrink monotonically throughout self-correction, the final retained pure region is contained in every preceding search region:

$$(\beta_i)^{*,(S)} \subseteq (\beta_i)^{(s)}, \qquad s = 0, \ldots, S. \tag{63}$$

Consequently, the theorem's preservation condition can be satisfied by taking $(\beta_i)^* = (\beta_i)^{*,(S)}$, allowing its finite-convergence argument to be applied even when intermediate boundary updates remove portions of the initial pure region, subject to the remaining assumptions of the theorem.

Nevertheless, the final retained pure region $(\beta_i)^{*,(S)}$ may be substantially smaller than the initial region $(\beta_i)^{*,(0)}$. This represents an inherent limitation of the refinement process: because the ground-truth labels of the unlabeled pool are unavailable, the true extent of the pure region cannot be determined directly. Consequently, boundary updates may conservatively exclude some pure samples to reduce impurity. This trade-off motivates the skipping strategy introduced in Section 4, which mitigates the limitations of excessively restrictive safe regions.

H.2 Further Discussions

Limitations. The proposed SafeHerding method and its underlying safe zone framework incur additional computational overhead compared with existing approaches (see Appendix F.4). Furthermore, the self-correction procedure may partially exclude initially pure regions, resulting in smaller retained regions that may still contain some noisy samples (see Section 3 and Appendix H.1). This limitation arises from the difficulty of accurately identifying pure regions without access to ground-truth labels. We therefore consider the development of reliable methods for estimating and preserving pure regions in the data space an important direction for future research.

Broader impacts. This work represents a theoretical advancement in the field of active learning. There could be some potential societal consequences of our work. For example, our work can be implemented in deep learning domains such as healthcare, autonomous driving, and recommendation systems, where labeled data are expensive or difficult to obtain.

H.3 Pseudocode of SafeHerding

In this section, we present the proposed SafeHerding framework in detail and provide its corresponding pseudocode. Algorithm 1 outlines the procedure of SafeHerding. At each active learning round t, we first apply the parameter adaptation mechanism of UHerding (Bae et al., 2025) and initialize the query set $\mathcal{Q}_t$ (lines 4-7). During the initial stage, when fewer than M labeled samples are available, SafeHerding follows the original uncertainty coverage objective (lines 13-14). Once $|\mathcal{L}_t| \geq M$, SafeHerding leverages the first M labeled samples as the centroids. Then the method constructs the search regions and the corresponding safe zone Ω_{safe} following Section 3.1 (lines 8-9). SafeHerding then selects $\mathcal{B}_t$ samples sequentially by maximizing safe coverage (lines 15-16). After achieving the query set $\mathcal{Q}_t$, we obtain the ground-truth labels of $\mathcal{Q}_t$ from the oracle f. Then the self-correction mechanism in Section 3.2 is applied on incorrect samples in $\mathcal{Q}_t$, adjusting the corresponding boundary and updating the safe zone (lines 22-24). The queried samples are subsequently added to $\mathcal{L}_{t+1}$, removed from $\mathcal{U}_{t+1}$, and used to update the learner $\hat{f}$ (lines 26-27). SafeHerding then evaluates the agreement score ϕ_i for each search region (line 31). Regions satisfying $\phi_i \geq \lambda$ are removed from the safe-zone (lines 32-34). Furthermore, when the global agreement score satisfies $\bar{\phi} \geq \Lambda$ or the size of safe zone Ω_{safe} is not enough, SafeHerding completely removes the safe-zone constraint and switches to the original uncertainty-coverage objective for all subsequent rounds (lines 36-43). Through these local and global skipping mechanisms, SafeHerding progressively transitions from purity-aware selection in the early learning stage to broader coverage-aware exploration as the learner becomes more reliable.

Algorithm 1 SafeHerding

Input: Initial labeled set $\mathcal{L}_0$, initial unlabeled pool $\mathcal{U}_0$, number of active learning rounds T, query budgets $\{\mathcal{B}_t\}_{t=0}^{T-1}$, learner $\hat{f}$, feature extractor g, number of search regions M, local agreement threshold λ, and global agreement threshold Λ.
Output: Final labeled set $\mathcal{L}_T$ and remaining unlabeled pool $\mathcal{U}_T$.

Initialize `SafeConstructed` $\leftarrow$ `False`
Initialize `GlobalSkip` $\leftarrow$ `False`
for $t \in [0, 1, \ldots, T-1]$ **do**
  Apply the parameter adaptation mechanism of UHerding (Bae et al., 2025)
  $\mathcal{Q}_t \leftarrow \emptyset$
  $\widetilde{\mathcal{U}}_t \leftarrow \mathcal{U}_t$
  $\widetilde{\mathcal{L}}_t \leftarrow \mathcal{L}_t$
  if $|\mathcal{L}_t| \geq M$ **and** `SafeConstructed` $=$ `False` **then**
    Construct the search regions $\{(\beta_i)^{(t)}\}_{i=1}^{M}$ and the safe zone Ω_{safe} as described in Section 3.1
    `SafeConstructed` $\leftarrow$ `True`
  end if
  for $q \in [1, 2, \ldots, \mathcal{B}_t]$ **do**
    if `SafeConstructed` $=$ `False` **or** `GlobalSkip` $=$ `True` **then**
      Select
$$\mathbf{x}_q^* \in \arg\max_{\tilde{\mathbf{x}} \in \widetilde{\mathcal{U}}_t} \widehat{UC}_{\sigma_\gamma}\left(\widetilde{\mathcal{L}}_t \cup \mathcal{Q}_t \cup \{\tilde{\mathbf{x}}\}\right)$$
    else
      Select
$$\mathbf{x}_q^* \in \arg\max_{\tilde{\mathbf{x}} \in \widetilde{\mathcal{U}}_t} \widehat{SC}\left(\widetilde{\mathcal{L}}_t \cup \mathcal{Q}_t \cup \{\tilde{\mathbf{x}}\}\right)$$
    end if
    Update $\mathcal{Q}_t \leftarrow \mathcal{Q}_t \cup \{\mathbf{x}_q^*\}$
    Update $\widetilde{\mathcal{L}}_t \leftarrow \widetilde{\mathcal{L}}_t \cup \{\mathbf{x}_q^*\}$
    Update $\widetilde{\mathcal{U}}_t \leftarrow \widetilde{\mathcal{U}}_t \setminus \{\mathbf{x}_q^*\}$
  end for
  if `SafeConstructed` $=$ `True` **and** `GlobalSkip` $=$ `False` **then**
    Apply the self-correction procedure with $\mathcal{Q}_t$ in Section 3.2
    Update the corresponding search region and Ω_{safe}
  end if
  Update $\mathcal{L}_{t+1} \leftarrow \mathcal{L}_t \cup \mathcal{Q}_t$
  Update $\mathcal{U}_{t+1} \leftarrow \mathcal{U}_t \setminus \mathcal{Q}_t$
  Train the learner $\hat{f}$ using $\mathcal{L}_{t+1}$
  if `SafeConstructed` $=$ `True` **and** `GlobalSkip` $=$ `False` **then**
    for $i = 1, 2, \ldots, M$ **do**
      Compute the agreement score ϕ_i of $(\beta_i)^{(t)}$ using Eq. (16)
      if $\phi_i \geq \lambda$ **then**
        $\Omega_{\text{safe}} \leftarrow \Omega_{\text{safe}} \setminus (\beta_i)^{(t)}$
      end if
    end for
    Compute $\bar{\phi} \leftarrow \dfrac{1}{M}\sum_{i=1}^{M} \phi_i$
    if $\bar{\phi} \geq \Lambda$ **then**
      `GlobalSkip` $\leftarrow$ `True`
    end if
  end if
  if $|\Omega_{\text{safe}}| < \mathcal{B}_t$ **then**
    `GlobalSkip` $\leftarrow$ `True`
  end if
end for
return $\mathcal{L}_T, \mathcal{U}_T$