ACTIVE LEARNING FOR CONDITIONAL FLOW MATCHING MODEL IN ENGINEERING DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Although the flow matching model has demonstrated powerful capabilities in modern machine learning, its training notoriously relies on an incredibly large scale of high-quality labeled samples. Nevertheless, the acquisition of high-quality labeled datasets is hindered by exorbitant labeling costs in certain fields, notably engineering design. Therefore, selecting the most informative samples for training at minimal cost poses a key challenge in this field. This issue constitutes a central topic in active learning, a subfield of machine learning dedicated to maximizing model performance while minimizing annotation cost. The central challenge involves developing a query strategy to acquire the most informative data samples with minimal labeling effort. This paper presents a pilot study that investigates the application of active learning, which traditionally explored within the context of discriminative models, to flow matching models. By analyzing conditional flow matching through a piecewise-linear closed-form formulation, this work provides insight into how individual data points can affect the diversity and conditional accuracy of the resulting generative model. Building on this analysis, we propose two distinct query strategies: one aimed at enhancing model diversity, and the other designed to improve model accuracy. These strategies highlight a dataset-level trade-off between diversity and accuracy, and motivate a hybrid query strategy that balances the two objectives. Extensive experiments validate the effectiveness of our approach, showing that the proposed query strategies outperform baselines in terms of diversity and accuracy, respectively.

1 INTRODUCTION

Recently, flow matching models achieve state-of-the-art performance in image and various other generating tasks (Dhariwal & Nichol (2021); Ho et al. (2022); Saharia et al. (2022)) and are one of the fundamental building blocks of the more advanced image and video synthesis systems, e.g., DALL-E-3 (Ramesh et al. (2022)) and Veo3 (Esser et al. (2023)). The success of these models is attributed primarily to the availability of large-scale, high-quality labeled training datasets.

However, the acquisition of high-quality labeled datasets is notoriously challenging in some domains due to exorbitant annotation costs. This is particularly true in fields like medical imaging Budd et al. (2021) and engineering design Wu et al. (2024), where the cost of obtaining labels far exceeds that of data acquisition (A brief introduction to the engineering design is shown in the Appendix A.1). For instance, in medical imaging, the cost of annotating images by expert radiologists significantly exceeds the initial image acquisition cost. Similarly, in engineering design, while drafting raw models is relatively inexpensive, obtaining high-fidelity automotive performance results, which require extensive physics experiments and expert interpretation, entails substantially greater effort and expense. For example, Computational Fluid Dynamics simulation for airfoil requires about 25 minutes per sample Bonnet et al. (2022). Wind tunnel testing requires $5,000-20,000 per configuration (automotive engineering). So a fundamental challenge in these fields is to select the most informative samples for labeling while minimizing cost. This problem defines the core mission of active learning, a machine learning subfield dedicated to maximizing model performance under constrained annotation resources by developing optimal query strategies.

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.

1.1 ACTIVE LEARNING

The most common Active Learning strategies include uncertainty based sampling Ren et al. (2021); Li et al. (2024a), query by Committee Seung et al. (1992), and representation-based sampling Geifman & El-Yaniv (2017); Sener & Savarese (2017), etc. The core principle guiding these methods is to identify and query the most valuable samples to improve the model's decision boundary. Meanwhile, a parallel research direction explores the integration of generative models within the active learning framework. For example, GAAL Zhu & Bento (2017); Lan et al. (2024) proposes employing generative networks for data augmentation. However, its randomly generated samples do not necessarily yield higher informativeness than those in the original dataset. In contrast, BGADL Tran et al. (2019) simultaneously trains both a generative network and a classifier to produce samples within uncertain or disagreement regions. Subsequent methods, including VAAL Sinha et al. (2019) and TAVAAL Kim et al. (2021), further extended this concept by leveraging adversarial learning frameworks to enhance data augmentation and improve feature representation. However, these methods primarily focus on *"Generative models for active learning"*, rather than *"Active learning for generative models"*. In other words, their main objective is to boost the performance of discriminative models. Consequently, active learning specifically designed for generative models has received limited attention.

In addition, some existing works claimed to be designed for the diffusion model, but in fact naively extended active learning strategies tailored for discriminative models to generative settings, often without accounting for their fundamental differences Du & Li (2023); Lan et al. (2024); Antoniuk et al. (2025); Hassan et al. (2025). For example, UEADA Du & Li (2023) employs a classification diffusion model as a proxy to estimate uncertainty, subsequently selecting unlabeled samples with high uncertainty for annotation. ALIM Lan et al. (2024) employs kernel PCA followed by a chi-squared test on proxy feature representations to identify global statistical outliers, subsequently selecting unlabeled samples located in low-density regions. ISPZhang et al. (2024) considers "subject of interest" which transforming the open querying problem in the label space into a semi-open one. Specifically, they design and test algorithms on a set of specific labels rather than in the entire label space. EMGM Antoniuk et al. (2025) selects unlabeled samples associated with the specific proxy-predicted label. In sum, these methods do not design query methods from the perspective of conditional flow matching model itself. In this paper, we will introduce query methods designed from the characteristics of conditional flow matching model.

1.2 FLOW MATCHING MODEL

As an advanced generative framework Lipman et al. (2022), the flow matching model is widely utilized in tasks spanning image generation Ramesh et al. (2022) to engineering design Lin et al. (2025). The evaluation of these models commonly employs *Diversity* and *Accuracy* as the primary criteria Li et al. (2024b). These two metrics evaluate different aspects: *Diversity* ensures the model generates a wide variety of samples rather than converging to a few repetitive modes, while *Accuracy* ensures that the generated samples faithfully satisfy the predefined target labels or conditions.

In this paper, we focus on the generation results across the entire condition space rather than under specific conditions. To conduct such analysis, we propose an analysis framework based on piecewise-linear neural networks theory Montúfar et al. (2014); Goujon et al. (2024) and closed-form flow matching model Scarvelis et al. (2023); Chen (2025), which helps us analyze the generalization mechanism of conditional flow matching models. Specifically, we assume the conditional flow matching model's neural network is piecewise-linear (piecewise-linear is non-linear). Based on the generalization mechanism, we obtain the pattern of how data affects diversity and accuracy. Our analysis suggests that data with the same label in the dataset can contribute to the diversity of the model, while data with different labels in the dataset can contribute to the accuracy of the model. These findings provide insight into the diversity-accuracy trade-off associated with dataset composition. Guided by this insight, we formulate two active learning query strategies designed to augment diversity and accuracy individually. Furthermore, we demonstrate that a weighted integration of these complementary strategies provides a practical means to navigate this trade-off and balance both performance metrics.

The key contributions of this work are summarized as follows:

1. **Conditional Flow Matching Model Analysis Framework**: We introduce an analytical framework for conditional flow matching models that leverages piecewise-linear neural network theory and closed-form flow matching model theory, enabling theoretical characterization. This framework provides insight into how individual data points influence the model's diversity and accuracy.

2. **Efficient Query Strategy for Active Learning**: Leveraging the proposed analytical framework, we present a pilot study on the application of active learning to conditional flow matching models, introducing two complementary query strategies: one aimed at enhancing model diversity and the other at improving model accuracy. These strategies represent competing objectives, highlighting the trade-off between diversity and accuracy from a data-centric perspective.

3. **Experimental Validation**: Experiments on multiple datasets including a synthetic dataset and three real-world engineering design tasks show that the two strategies improve their respective target metrics compared with representative baselines. Additionally, a weighted combination of the two strategies can be formed to create a hybrid query strategy, allowing for a tunable trade-off between diversity and accuracy by adjusting the corresponding weights.

2 METHODOLOGY

2.1 PROBLEM DEFINITION

As presented in Appendix A.2, in the pool-based active learning method, we define $U^n = \{\boldsymbol{X}, \boldsymbol{Y}\}$ as an unlabeled dataset with n samples where $\mathbf{x} \in \boldsymbol{X}$, $\mathbf{y} \in \boldsymbol{Y}$ ($\mathbf{y}$ and $\boldsymbol{Y}$ are unknown before labeled). $L^m = \{\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}\}$ is the current labeled dataset set with m samples, where $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$, $\boldsymbol{y} \in \boldsymbol{\mathcal{Y}}$. Our goal is to design a query strategy Q_D ($U^n \xrightarrow{Q_D} L^m$) to maximize the *Diversity* of the model, and a design query strategy Q_A ($U^n \xrightarrow{Q_A} L^m$) to maximize the *Accuracy* of the model in the limited budget b. It is worth noting that during the active learning process, we train the model exclusively on queried samples from the dataset, without incorporating any samples synthesized by generative models.

2.2 PIECEWISE-LINEAR ANALYSIS FRAMEWORK

In this paper, we leverage specific characteristics of neural networks to analyze the conditional flow matching model, rather than analyzing the complex networks themselves. Particularly, this investigation combines the piecewise-linear neural networks theory (CPWL NNs)Montúfar et al. (2014); Goujon et al. (2024) and the closed-form flow matching model theoryScarvelis et al. (2023).

The fundamental idea of CPWL NNs theory is that neural networks can be mathematically characterized as continuous piecewise-linear (CPWL) functions. This implies that their input domain can be partitioned into a finite set of closed, convex subsets (regions). Within each region, the network behaves as a simple affine function. This precise definition of a "linear region", based on partitioning the input space into non-overlapping, closed, convex subsets, is what allows us to describe a function that is **locally affine** in each region, despite being **globally-nonlinear**.

Closed-form flow matching model Scarvelis et al. (2023); Chen (2025) is the optimal solution of the flow matching model under the MSE loss function, which can explicit give the flow field of the flow matching model under different noise and t inputs (A brief derivation of the closed-form flow matching model is shown in the Appendix A.3). However, existing closed-form flow matching models only discuss unconditional flow fields. In this paper, we will discuss conditional flow fields.

Our analysis assumes a continuous piecewise-linear network and a closed-form flow matching model. Although highly expressive deep networks may not strictly satisfy these idealized conditions, the derivation provides geometric intuition for why label space sampling balances diversity and accuracy, serving as a conceptual motivation rather than a hard constraint on the trained model. We validate this intuition empirically in Appendix H, where both assumptions are explicitly violated and the proposed strategies remain effective.

The theory should therefore be read as providing intuition and design principles; the empirical validation in Section 3 confirms that these principles hold in practice.

Specifically, when condition $\boldsymbol{c}_0$ is in the labels of the dataset, the flow field of the closed-form flow matching model is (A brief proof is shown in the Appendix A.3):

$$\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_0) = \frac{\sum_i^m p_{t,i} \boldsymbol{e}_{t,i}}{\sum_i^m p_{t,i}} \tag{1}$$

where $\boldsymbol{e}_{t,i} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ is the noise that make $\boldsymbol{x}_i$ to $\boldsymbol{x}'$ ($\boldsymbol{x}' = (1-t)\boldsymbol{e}_{t,i} + t\boldsymbol{x}_i$), $\boldsymbol{x}_i$ is the data with label $\boldsymbol{c}_0$ in the dataset, m is the number of the data with label $\boldsymbol{c}_0$ in the dataset, $p_{t,i}$ is the probability density of $\boldsymbol{e}_{t,i}$. Equation (1) means the vector field is a linear combination of data in a dataset.

When condition $\boldsymbol{c}^* \in \mathbb{R}^d$ is not in the labels of the dataset, we apply the CPWL assumption. The label space is divided into several non-overlapping linear sub regions by the labels in the dataset (which can be a Delaunay Subdivision or not, but this does not affect the subsequent derivation), each sub region being a convex hull (simplex) with $d+1$ vertices (labels in dataset). This means that $\boldsymbol{c}^*$ must fall within one of the linear subregions. The output of the neural network is defined as the interpolation of the output of the neural network given the conditions at vertices of this sub region:

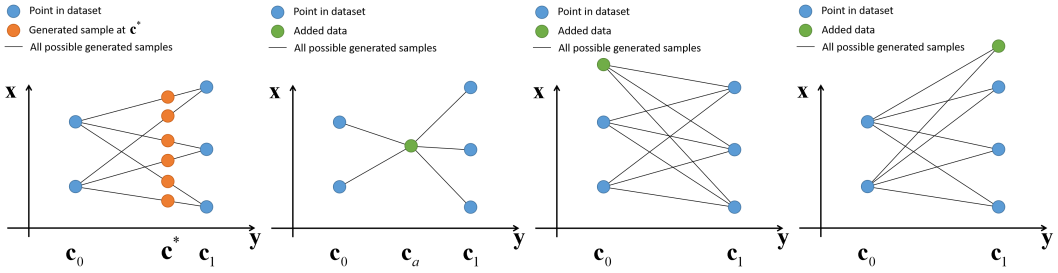
$$\boldsymbol{u}_t(\boldsymbol{x}', a_0\boldsymbol{c}_0 + a_1\boldsymbol{c}_1 + ... + a_d\boldsymbol{c}_d) = a_0\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_0) + a_1\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_1) + ... + a_d\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_d) \tag{2}$$

where $\boldsymbol{c}^* = a_0\boldsymbol{c}_0 + a_1\boldsymbol{c}_1 + ... + a_d\boldsymbol{c}_d$. a_0, a_1, and a_d are interpolation coefficients. $\boldsymbol{c}_0$, $\boldsymbol{c}_1$ and $\boldsymbol{c}_d$ are the labels that form sub region vertices in the dataset. $[a_0, a_1, ..., a_d]$ can be easily calculated using the label $[\boldsymbol{c}_0, \boldsymbol{c}_1, ..., \boldsymbol{c}_d]$. Because $d+1$ points form a d-dimensional plane.

Under this assumption, when the given condition $\boldsymbol{c}_0$ exists in the dataset, the closed-form flow matching model is constrained to output samples with label $\boldsymbol{c}_0$ in the dataset Scarvelis et al. (2023).

Besides, when the given condition $\boldsymbol{c}^*$ does **not** exists in the dataset, by using Lemma B.1 (proven in Appendix B), we know that the vector field in Equation (2) will result in the generated sample $\boldsymbol{x}^*$, being an interpolation of $[\boldsymbol{x}_i, \boldsymbol{x}_j, ..., \boldsymbol{x}_k]$. ($\boldsymbol{x}_i$ stands for any sample labeled $\boldsymbol{c}_0$ in the dataset, $\boldsymbol{x}_j$ stands for any sample labeled $\boldsymbol{c}_1$ in the dataset, etc.)

$$\{\boldsymbol{x}^* | \boldsymbol{x}^* = a_0\boldsymbol{x}_i + a_1\boldsymbol{x}_j + ... + a_d\boldsymbol{x}_k\} \tag{3}$$



(a) Generated samples at $\boldsymbol{c}^*$(b) Adding new point at $\boldsymbol{c}_a$.(c) Adding new point at $\boldsymbol{c}_0$.(d) Adding new point at $\boldsymbol{c}_1$.

Figure 1: Comparison of different point addition strategies and their all possible generated samples ($\boldsymbol{c} \in \mathbb{R}^1$), with the black line indicating the possible generated samples.

Equation (3) provides the generation law of the piecewise-linear closed-form flow matching model. Specifically, interpolation in the label space results in corresponding interpolation in the data space as illustrated in Figure 1a. It is worth noting that the label dimension d is generally smaller than the data dimension, meaning that the interpolation coefficients derived from the labels induce lower-dimensional interpolation in the data space (A simple experimental demonstration of this phenomenon is included in Appendix C). As a methodological note, the generated samples are provided without accounting for their respective generation probabilities. The probability of certain samples can be very small and even zero, as it is inherently contingent on the input condition, labels, and the characteristics of the data distribution. Therefore, Equation (3) establishes an upper bound on the diversity of generated samples. Notably, the maximum number of units for d-dimensional simplex partitioning is $O(n^{\lceil d/2 \rceil})$, n is the number of points. For a deep network that uses a total of D hidden units with ReLU activation, the upper limit of the number of linear regions is $O(2^D)$. However, we

do not directly calculate these partitions, we only use them in deriving active learning algorithms. Therefore, there will be no curse of dimensionality increasing with d.

2.3 DIVERSITY FROM DATASET

Building upon the analysis in the previous subsection, we have derived the generation rules of the conditional flow matching model. This understanding facilitates the analysis of how specific data points influence both the diversity and accuracy of a trained conditional flow matching model.

Building on Dombrowski et al. (2025), who quantify diversity by the number of samples a model can retrieve from a dataset, we offer a dual perspective: the diversity of a generative model can be enhanced by maximizing the cardinality of the set of distinct samples it can generate under a given condition $\boldsymbol{c}^*$.

To conclude, our analysis of the piecewise-linear closed-form flow matching model suggests that the three identified data patterns contribute to enhancing model diversity.

- **Pattern 1**: Data sharing the same label enhance the model's diversity.
- **Pattern 2**: Diversity increases when data points are more evenly distributed across clusters (Clusters are defined as groups of data points with the same label).
- **Pattern 3**: The greater the distance between new data point and existed data points, the greater the diversity.

For the sake of simplicity, consider the case of $\boldsymbol{c} \in \mathbb{R}^1$ and $d = 1$. As shown in Figure 1a, when $\boldsymbol{c}^* \in (\boldsymbol{c}_0, \boldsymbol{c}_1)$, and there are m samples labeled as $\boldsymbol{c}_0$ and n samples labeled as $\boldsymbol{c}_1$ in the dataset (no data labeled between $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$), the maximum generated sample type under condition $\boldsymbol{c}^*$ is mn.

As shown in Figure 1b, when adding a new data with label $\boldsymbol{c}_a$ ($\boldsymbol{c}_0 < \boldsymbol{c}_a < \boldsymbol{c}_1$) to the dataset, the interval $(\boldsymbol{c}_0, \boldsymbol{c}_1)$ is divided into two segments $(\boldsymbol{c}_0, \boldsymbol{c}_a)$ and $(\boldsymbol{c}_a, \boldsymbol{c}_1)$. When $\boldsymbol{c}^* \in (\boldsymbol{c}_0, \boldsymbol{c}_a)$, the model will generate up to m types of samples, and when $\boldsymbol{c}^* \in (\boldsymbol{c}_a, \boldsymbol{c}_1)$, the model will generate up to n types of samples. Compared to the original dataset, this point adding strategy reduces the number of types of points at each $\boldsymbol{c}^*$, thereby decreasing the diversity of the model. Therefore, to increase the diversity of the model, we can only consider adding data labeled $\boldsymbol{c}_0$ or $\boldsymbol{c}_1$.

As shown in Figure 1c, adding data points labeled $\boldsymbol{c}_0$ will result in the model generating up to $(m+1)n$ types of samples under condition $\boldsymbol{c}^* \in (\boldsymbol{c}_0, \boldsymbol{c}_1)$. While as shown in Figure 1d, adding data points labeled $\boldsymbol{c}_1$ will result in the model generating up to $m(n+1)$ types of samples under condition $\boldsymbol{c}^* \in (\boldsymbol{c}_0, \boldsymbol{c}_1)$. Obviously, to increase the number of types of points, we need to balance the number of data labeled $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$ in the dataset. In addition, when previous preceding patterns hold, adding data points that are far away from labeled dataset can increase the diversity defined by distance Friedman & Dieng (2022), even without increasing the types of generated samples.

Guided by the aforementioned analysis, we propose a query strategy Q_D. It selects unlabeled data points $\mathbf{x}$ form unlabeled pool $\boldsymbol{X}$ and augments the labeled dataset $\{\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}\}$, thereby increasing model diversity.

$$Q_D = \underset{(\mathbf{x},\mathbf{y})\in\{\boldsymbol{X},\boldsymbol{Y}\}}{\arg\max} \; -\alpha\mathrm{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}}) + \beta\Delta\mathrm{clusterentropy} + \gamma\mathrm{distance}(\mathbf{x}, \boldsymbol{\mathcal{X}}) \tag{4}$$

where α, β, γ, are weighting coefficients. $\mathrm{distance}$ means distance from unlabeled data point to labeled dataset, we chose the minimum Euclidean distance in the experiments. Specifically, the minimum distance between a data point and all points in the dataset. $\Delta\mathrm{clusterentropy}$ means the cluster entropy increase of labels brought by new labels.

The Q_D comprises **3 terms**, which align with the **3 Patterns** for diversity enhancement discussed earlier. The first term, $-\mathrm{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}}) = -\min_{\boldsymbol{y}\in\boldsymbol{\mathcal{Y}}} ||\mathbf{y} - \boldsymbol{y}||_2$, is designed to encourage new data points to have labels similar to those in the existing dataset. The preceding analysis prescribes that the labels of new data must be strictly identical to those already in the dataset. However, obtaining such exact matches is typically infeasible in practice. Accordingly, we impose the weaker condition that the labels of new data exhibit sufficient similarity to the labels present in the dataset. In fact, since neural networks are continuous, they can recognize similar labels as the same label without the need

for complete identity. For unlabeled data, we employ Radial Basis Function (RBF) Neural Networks for label prediction due to their favorable optimization properties and computational efficiency.

The second term, $\Delta\text{clusterentropy}$, is designed to encourage the selection of new data points that promote a more balanced distribution in the quantity of data in different cluster. This cluster entropy corresponds to classification cluster entropy $H = -\sum p_k \log p_k$ (p_k is proportion of samples in cluster k), rather than being computed directly as information entropy. Specifically, we first partition the dataset labels into clusters and then compute the cluster entropy of the label distribution across these clusters. In practical use, a cluster is defined as a set of labels whose inter-point distances fall below a given threshold τ.

The last term, $\text{distance}(\mathbf{x}, \boldsymbol{\mathcal{X}}) = \min_{\boldsymbol{x} \in \boldsymbol{\mathcal{X}}} ||\mathbf{x} - \boldsymbol{x}||_2$, is designed to encourage the query strategy to select new data points that are farther from the existing dataset once the first two conditions are satisfied, thereby avoiding duplicating data and improving diversity.

2.4 Accuracy from Dataset

Through the analysis in Section 2.2, it can be concluded that interpolation in the condition space leads to corresponding interpolation in the data space. Furthermore, Lemma B.2 (proven in Appendix B) provides the error bound of the model within a subregion, given by:

$$|f(\boldsymbol{x}^*) - \boldsymbol{c}^*| \leq K \max ||\boldsymbol{c}_i - \boldsymbol{c}_j||^2 \tag{5}$$

where $f(\boldsymbol{x}) = \boldsymbol{y}$ represents authentic labels, K is related to f and d. $\boldsymbol{c}^*$ is the target condition, $\boldsymbol{x}^*$ is the sample generated by the model given $\boldsymbol{c}^*$. $max||\boldsymbol{c}_i - \boldsymbol{c}_j||^2$ means the maximum distance of any two points in the subregion of label space.

In Equation (5), the upper bound on the error within each subregion is determined by the maximum distance between any two points in the subregion. To reduce the error upper bound, a natural approach is to minimize this maximum distance. Accordingly, within the query strategy aimed at enhancing model accuracy, it is intuitive to select new data points whose labels are farthest from those already present in the dataset, as illustrated in Equation (6). Essentially, Q_A performs a coreset-like algorithm Sener & Savarese (2017) in the label space.

$$Q_A = \underset{(\mathbf{x},\mathbf{y}) \in \{\boldsymbol{X}, \boldsymbol{Y}\}}{\arg\max} \ \text{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}}) \tag{6}$$

Upon comparing Equation (4) and Equation (6), it becomes apparent that the two strategies exhibit a fundamental conflict: Q_D aims to seek new samples with $\text{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}})$ being smaller, while Q_A aims to seek new samples with $\text{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}})$ being larger. In other words, data sharing the same label enhance the model's diversity, whereas data with distinct labels improve its accuracy. This clarifies why diversity and accuracy represent a trade-off from the perspective of dataset composition.

Considering that Equation (4) solely enhances model diversity while Equation (6) only improves model accuracy, a natural extension is to combine these two query strategies to balance the trade-off between diversity and accuracy. This leads to:

$$Q_{Hybrid} = \omega Q_D + (1 - \omega) Q_A \tag{7}$$

where $\omega \in [0, 1]$ controls the ratio of Q_D to Q_A.

Figure 2 presents the query results of different query methods. As shown in Figure 2, the dataset is unevenly distributed in both the data space and the label space. Different query strategies lead to the selection of different new data points. In particular, the CoreSet method selects data points that ensure a larger coverage of the feature space. The Committee method selects new data with the greatest output discrepancy among prediction models. Consequently, the selected samples tend to cluster at the edges of the label distribution, such as the regions corresponding to labels 0 and 1. This divergence arises from the distinct extrapolation strategies employed by the different prediction models. Q_D selects new data by ensuring a uniform distribution of labels across different clusters in the label space, while simultaneously maximizing the distance from the initial data in the data space. Q_A selects new data such that the labels of the data are uniformly distributed across the label space.

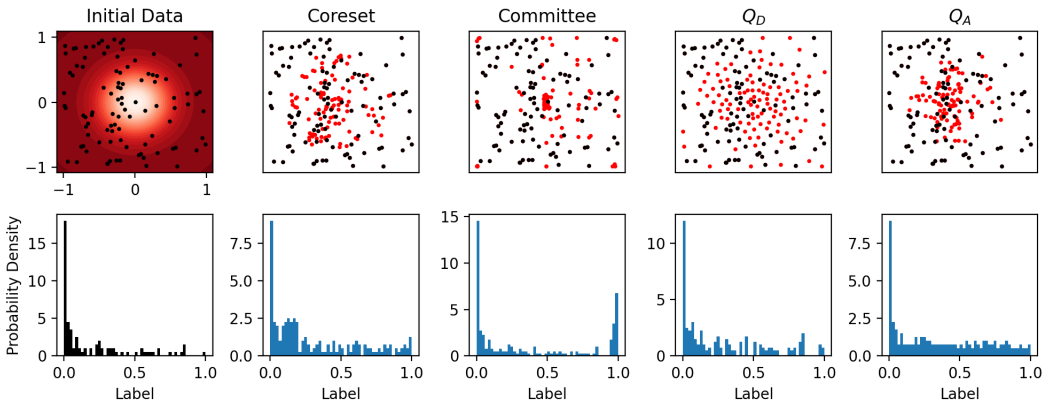


Figure 2: Comparison of different query strategies in Uneven dataset. Three-quarters of the data points are evenly located on the left side, while the remaining quarter are on the right. The label $\boldsymbol{y} = f(\boldsymbol{x})$ is the Gaussian distribution with $\boldsymbol{\mu}$ at $[0, 0]$. The first row depicts the data distributions, and the second row shows the corresponding label distributions. In all subfigures, the black points represent the initial data, while the red points represent the new data selected by each query strategy.

3 Experiment

3.1 Dataset and Metrics

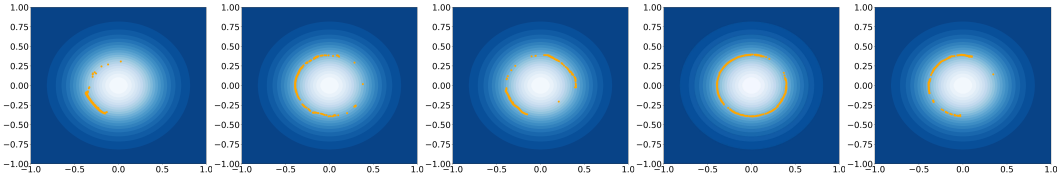
We conducted experiments across four datasets to validate our algorithm. These datasets comprises one toy dataset (Uneven dataset) for conceptual verification and three real-world design tasks (Airfoil dataset, Flying-Wing dataset, Starship dataset) for practical assessment. The detailed dataset introduction is shown in the Appendix D.1 The geometric models, such as airfoils, flying-wings, and starships, are readily available; however, acquiring their corresponding labels (performances) necessitates enormous costs Wu et al. (2024).

The conditional flow matching model is built in the parametric feature space of these shapes. Our evaluation framework is designed to measure diversity and accuracy separately, rather than using a combined metric such as FID Yu et al. (2021). Diversity is quantified by a custom variant of the Vendi score Friedman & Dieng (2022), calculated as the average pairwise distance such as Euclidean distance (L2) and similarity of the generated data points. Accuracy is evaluated by measuring the discrepancy between the true labels of generated samples and the specified target conditions, using Mean Squared Error (MSE) and Mean Absolute Error (MAE). Crucially, the true labels are computed by a simulation solver applied to each generated geometry. This evaluation protocol inherently enforces physical feasibility: if a generated shape violates physical constraints, for instance by containing negative-volume cells, the solver returns NaN. Consequently, any sample that receives a valid numerical label from the solver is guaranteed to be physically realizable, providing an implicit yet rigorous measure of design feasibility.

$$diversity\ score = \int_{\mathbb{Y}} \mathbb{E}||\boldsymbol{x}_{gen,i} - \boldsymbol{x}_{gen,j}||_2 \, d\boldsymbol{c} \tag{8}$$

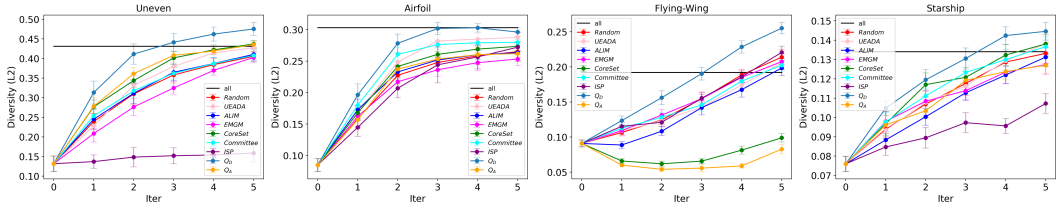
$$accuracy\ score = \int_{\mathbb{Y}} \mathbb{E}(\boldsymbol{c} - \boldsymbol{y}_{gen,i})^2 \, d\boldsymbol{c} \tag{9}$$

where $\boldsymbol{x}_{gen}$ denotes a generated sample, $\boldsymbol{y}_{gen}$ denotes its corresponding label. Conceptually, both the diversity score (Equation (8)) and accuracy score (Equation (9)) are defined directly on the label space $\mathbb{Y}$, within which the Riemann integration is performed for evaluation.

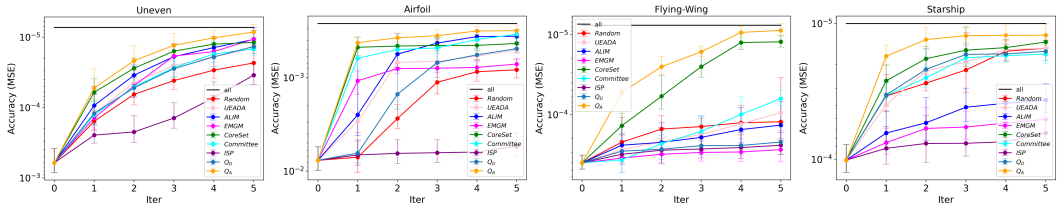


(a) Initial Data (b) CoreSet (c) Committe (d) Q_D (e) Q_A

Figure 3: Comparison of samples generated given condition 0.5 in the Uneven dataset.



(a) Comparison of diversity on four datasets (higher line is better).



(b) Comparison of accuracy on four datasets (higher line is better).

Figure 4: Comparison of diversity (L2) and accuracy (MSE) on four datasets (mean and std). For clarity, vertical coordinates of Accuracy have been reversed, making higher lines correspond to better model performance.

3.2 Results

In each iteration of these experiments, 6% of the data is selected and labeled. Data with labels is used for conditional training, while all the data is used for unconditional training. The initial (0-th) round of data selection is performed randomly with same random seed for all methods, yielding identical start results. In our experiments, we conducted runs across twenty random seeds. The reported results correspond to the mean and standard deviation calculated over these seeds. While evaluating model performance requires producing samples via a generative model, the training process relies solely on the queried samples. Specifically, no shapes generated by any trained generative models are incorporated into the training set. In this section, diversity score in Equation (8) is calculated with L2 and accuracy score in Equation (9) is calculated with MSE. More results including time testing Appendix D.2, generated results Appendix D.3, more baselines Appendix D.5, supplementary evaluation metrics Appendix D.4, ablation study Appendix D.6, hyperparameter and hyperparameter sensitivity Appendix E, statistical validation and analysis Appendix F, proxy error analysis Appendix G, and experiments under assumption violations Appendix H.

Figure 3 shows the samples generated by the model under different point selection strategies given condition 0.5 in Uneven dataset. The optimal generation result under condition 0.5 is a circle located at the origin. Figure 3a shows model trained on initial data points. It can be seen that due to insufficient data, even on the left half, the generated result is not a complete semicircle. Among all methods, Q_D has the highest diversity, while Q_A has the smallest diversity.

For the baselines, we select classic active learning algorithms designed for discriminative models, namely CoreSet Sener & Savarese (2017) and Committee, as well as those designed for generative models, including UEADA Du & Li (2023), ALIM Lan et al. (2024), ISP Zhang et al. (2024), and

EMGM Antoniuk et al. (2025). For the Committee method, SVR, Random Forest, XGBoost, and RBF neural networks are employed to predict the labels of unlabeled data points; the variance of their predictions is then used as the criterion for selecting new samples. Furthermore, we incorporate more advanced active learning algorithms for discriminative models, such as ProbCover Yehuda et al. (2022), MaxHerding Bae et al. (2024), UHerding Bae et al. (2025), BADGE Ash et al. (2019), and TypiClust Hacohen et al. (2022). For the sake of clarity, the results of these five advanced discriminative baselines are deferred to Appendix D.5.

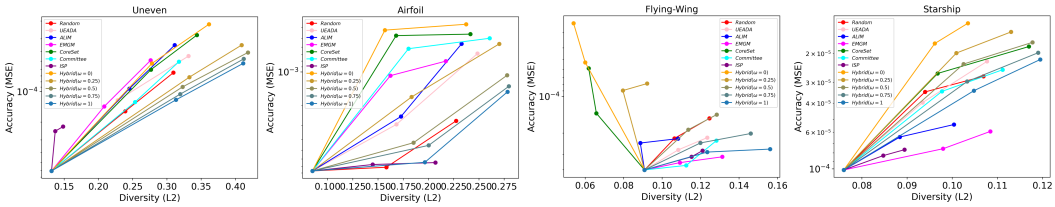


Figure 5: Comparison of diversity and accuracy with different ω (top right is better).

Figure 4 compares the diversity and accuracy across the four datasets. The results indicate that Q_D achieves the highest diversity in all datasets, even outperforming the model trained on the full dataset, although this comes at the cost of reduced accuracy. In contrast, Q_A yields the highest accuracy in all datasets with the cost of reduced diversity.

Notably, training on a partial subset can yield higher diversity than training on the full dataset. This is because, when the original dataset is imbalanced, a model trained on the full data tends to produce a more concentrated output distribution, thereby reducing overall diversity. In Figure 4a, the diversity of some methods on the Flying-Wing dataset decreases during iter 0-2, which is inconsistent with the monotonically increasing trend observed in other datasets. This phenomenon is reasonable because increasing the number of data points does not necessarily improve the diversity metric of the model, especially when using strategies to improve model accuracy. Figure 1b clearly indicates the case that increasing sample points may reduce the diversity metric of the model through the piecewise-linear closed-form flow matching analysis. Specifically, our analysis framework states that as the number of data points increases, accuracy remains non-decreasing.

Notably, using a partial dataset can surpass the entire dataset in diversity, but cannot surpass the entire dataset in accuracy. Because when the dataset is imbalanced, model trained on full dataset will generate a more concentrated data distribution, which has a lower diversity. While increasing data will increase the maximum coverage area of the label space and then increase accuracy.

Figure 5 illustrates how the weight ω in Equation (7) can be tuned to control the trade-off between diversity and accuracy: a larger ω prioritizes diversity, while a smaller ω favors accuracy. The Hybrid strategy roughly constitutes a Pareto front that is not inferior to the baseline. However, linear mixing does have room for improvement, we will consider a more efficient hybrid strategy in future work.

4 Conclusion

This work investigated active learning for conditional flow matching in engineering design. By analyzing a closed-form flow matching model with piecewise-linear networks, we derived geometric intuition for the distinct roles of labeled data: label-similar points promote generation diversity, while label-distant points enhance conditional accuracy. Motivated by this insight, we proposed two dedicated query strategies, one targeting diversity and the other targeting accuracy, along with a weighted hybrid that balances both objectives. Comprehensive experiments including a synthetic dataset and three real-world engineering design tasks confirm that our strategy surpasses baseline active learning strategies in terms of diversity and accuracy, respectively. Additionally, a weighted combination of these two strategies can be formed to create a hybrid query strategy, allowing for a tunable trade-off between diversity and accuracy by adjusting their weights. However, the hybrid strategy adopts a simple linear weighting method without exploring an adaptive adjustment mechanism based on training dynamics, which cannot quantitatively control this trade-off. We will consider a more efficient hybrid strategy in future work.

REFERENCES

Evan R Antoniuk, Peggy Li, Nathan Keilbart, Stephen Weitzner, Bhavya Kailkhura, and Anna M Hiszpanski. Active learning enables extrapolation in molecular generative models. *arXiv preprint arXiv:2501.02059*, 2025.

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.

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.

Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. *Advances in Neural Information Processing Systems*, 35:23463–23478, 2022.

Samuel Budd, Emma C Robinson, and Bernhard Kainz. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical image analysis*, 71:102062, 2021.

Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. *arXiv preprint arXiv:2209.11215*, 2022.

Zhengdao Chen. On the interpolation effect of score smoothing. *arXiv preprint arXiv:2502.19499*, 2025.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Mischa Dombrowski, Weitong Zhang, Sarah Cechnicka, Hadrien Reynaud, and Bernhard Kainz. Image generation diversity issues and how to tame them. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3029–3039, 2025.

Zhekai Du and Jingjing Li. Diffusion-based probabilistic uncertainty estimation for active domain adaptation. *Advances in Neural Information Processing Systems*, 36:17129–17155, 2023.

Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7346–7356, 2023.

Dan Friedman and Adji Bousso Dieng. The vendi score: A diversity evaluation metric for machine learning. *arXiv preprint arXiv:2210.02410*, 2022.

Yonatan Geifman and Ran El-Yaniv. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941*, 2017.

Alexis Goujon, Arian Etemadi, and Michael Unser. On the number of regions of piecewise linear neural networks. *Journal of Computational and Applied Mathematics*, 441:115667, 2024.

Guy Hacohen, Avihu Dekel, and Daphna Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets. *arXiv preprint arXiv:2202.02794*, 2022.

Sabit Hassan, Anthony B Sicilia, and Malihe Alikhani. An active learning framework for inclusive generation by large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 5403–5414, 2025.

Amin Heyrani Nobari, Wei Chen, and Faez Ahmed. Pcdgan: A continuous conditional diverse generative adversarial network for inverse design. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 606–616, 2021.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Kwanyoung Kim, Dongwon Park, Kwang In Kim, and Se Young Chun. Task-aware variational adversarial active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8166–8175, 2021.

Guipeng Lan, Shuai Xiao, Jiachen Yang, Jiabao Wen, Wen Lu, and Xinbo Gao. Active learning inspired method in generative models. *Expert Systems with Applications*, 249:123582, 2024.

Holden Lee, Jianfeng Lu, and Yixin Tan. Convergence of score-based generative modeling for general data distributions. In *International Conference on Algorithmic Learning Theory*, pp. 946–985. PMLR, 2023.

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, 2024a.

Shuangqi Li, Chen Liu, Tong Zhang, Hieu Le, Sabine Süsstrunk, and Mathieu Salzmann. Controlling the fidelity and diversity of deep generative models via pseudo density. *arXiv preprint arXiv:2407.08659*, 2024b.

Jia-hao Lin, Shu-sheng Chen, Shi-yi Jin, Quan-feng Jiang, Mu-liang Jia, and Dong Li. Multi-point aerodynamic inverse design of flying wing using the conditional diffusion model. *Physics of Fluids*, 37(7), 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Tao Luo, Zhi-Qin John Xu, Zheng Ma, and Yaoyu Zhang. Phase diagram for two-layer relu neural networks at infinite-width limit. *Journal of Machine Learning Research*, 22(71):1–47, 2021.

Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, 27, 2014.

Yurui Qian, Qi Cai, Yingwei Pan, Yehao Li, Ting Yao, Qibin Sun, and Tao Mei. Boosting diffusion models with moving average sampling in frequency domain. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8911–8920. IEEE, 2024.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE transactions on pattern analysis and machine intelligence*, 45(4):4713–4726, 2022.

Christopher Scarvelis, Haitz Sáez de Ocáriz Borde, and Justin Solomon. Closed-form diffusion models. *arXiv preprint arXiv:2310.12395*, 2023.

Erik Seedhouse. Starship. In *SpaceX: Starship to Mars–The First 20 Years*, pp. 171–188. Springer, 2022.

{Michael S.} Selig. Uiuc airfoil data site, 1996.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 287–294, 1992.

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5972–5981, 2019.

Toan Tran, Thanh-Toan Do, Ian Reid, and Gustavo Carneiro. Bayesian generative active deep learning. In *International conference on machine learning*, pp. 6295–6304. PMLR, 2019.

Yueqing Wang, Peng Zhang, Yushuang Liu, Jianing Zhao, Jie Lin, and Yi Chen. Aerodynamic coefficients prediction via cross-attention fusion and physical-informed training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 869–876, 2025.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024.

Zhi-Qin John Xu, Yaoyu Zhang, and Zhangchen Zhou. An overview of condensation phenomenon in deep learning. *arXiv preprint arXiv:2504.09484*, 2025.

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.

Yu Yu, Weibin Zhang, and Yun Deng. Frechet inception distance (fid) for evaluating gans. *China University of Mining Technology Beijing Graduate School*, 3(11), 2021.

Xulu Zhang, Wengyu Zhang, Xiaoyong Wei, Jinlin Wu, Zhaoxiang Zhang, Zhen Lei, and Qing Li. Generative active learning for image synthesis personalization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 10669–10677, 2024.

Jia-Jie Zhu and José Bento. Generative adversarial active learning. *arXiv preprint arXiv:1702.07956*, 2017.

A CONCEPT INTRODUCTION

Considering that some readers may not be familiar with some basic concepts in this paper, we will introduce them in this section, including the background introduction, the theories not proposed by us but cited by us, and a perspective of generalization.

A.1 ENGINEERING DESIGN

Engineering design is one of the application of generative models Heyrani Nobari et al. (2021). When the user gives a specified target performance, the generative model provides a new shape that should have the target performance. For example, a user requires a car with a fuel consumption of $10L/100KM$ (target performance) on the urban road (working condition), and the model outputs a car with a fuel consumption of $10L/100KM$ on the urban road. Specifically, engineering design as a domain for generative models is characterized by labels that are defined over a continuous space, rather than a discrete set of categories.

A.2 ACTIVE LEARNING ALGORITHM

Algorithm 1 Active Learning Algorithm

Input: an unlabeled dataset $U = \{\boldsymbol{X}, \boldsymbol{Y}\}$, $\mathbf{x} \in \boldsymbol{X}$, $\mathbf{y} \in \boldsymbol{Y}$, labeled dataset $L = \{\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}\}$, $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$, $\boldsymbol{y} \in \boldsymbol{\mathcal{Y}}$, budget b, query strategy Q.
repeat
  $\mathbf{x}_{add} = Q(U, L)$
  $L = L \cup (\mathbf{x}_{add}, \mathbf{y}_{add})$
until Budget exhausted
Train model using the dataset (not generated samples)

A.3 CLOSED-FORM FLOW MATCHING MODEL

The closed-form diffusion model has been mentioned in many papers Karras et al. (2022); Chen et al. (2022); Lee et al. (2023), pointing out that the diffusion model has a solution that minimizes the loss function (calculated entirely using the dataset without training the neural network). That is, for any $\boldsymbol{x}_t$ and t input to the model and the model has an analytical output. As a special case of diffusion models, the flow matching model can also easily derive a closed form flow matching model.

Equation (1) can be directly derived from Equation 8 (Equation (10) in our paper) in classical flow matching papers Lipman et al. (2022).

$$\boldsymbol{u}_t(\boldsymbol{x}) = \int \boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_1)\frac{p_t(\boldsymbol{x}|\boldsymbol{x}_1)q(\boldsymbol{x}_1)}{p_t(\boldsymbol{x})}d\boldsymbol{x}_1 \tag{10}$$

Specifically, we can convert the integral of the original equation into a sum, because on a certain dataset, the probability distribution of the samples is composed of Dirac functions (each data point has a probability of $1/n$). By substituting the dataset into $\boldsymbol{x}_1$, we can obtain the Equation (1) in our paper.

A.4 ON METRIC FROM THE PERSPECTIVE OF GENERALIZATION

From a generalization perspective, we posit that generative models can be analyzed analogously to discriminative models through the lens of empirical error (E_{in}) and generalization error (E_{out}), with diversity and accuracy serving as critical performance indicators. Taking accuracy as a representative metric, the ideal objective is to achieve $E_{in} \approx E_{out} \approx 0$, ensuring robust performance on both training and unseen data, while minimizing the generalization gap ($E_{out} - E_{in}$) to signify strong generalization capability. Within the closed-form flow matching framework, the model theoretically achieves $E_{in} = 0$ on the training set; consequently, we utilize linear interpolation to estimate E_{out}, rendering the generalization gap equivalent to E_{out} ($E_{out} - E_{in}$) itself. Therefore, our active learning

strategy is fundamentally designed to minimize E_{out} through strategic sample selection, thereby maximizing generalization performance by reducing the error bound on unseen conditions to the greatest extent possible.

B MATHEMATICAL PROOF

The mathematical notation of the flow matching model: $\boldsymbol{x}_0 \in N(\mathbf{0}, \boldsymbol{I})$, $\boldsymbol{x}_1$ is data in dataset. $\boldsymbol{u}_t$ is the flow field at time t.

Lemma B.1. *In closed-form flow matching model, if* $\boldsymbol{u}_t(\boldsymbol{x}', a_0\boldsymbol{c}_0 + a_1\boldsymbol{c}_1 + ... + a_d\boldsymbol{c}_d) = a_0\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_0) + a_1\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_1) + ... + a_d\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_d)$*, then the flow matching model will generate the data in* $\{\boldsymbol{x}^*|\boldsymbol{x}^* = a_0\boldsymbol{x}_i + a_1\boldsymbol{x}_j + ... + a_d\boldsymbol{x}_k\}$ *when given* $\boldsymbol{c}^* = a_0\boldsymbol{c}_0 + a_1\boldsymbol{c}_1 + ... + a_d\boldsymbol{c}_d$*.* $\boldsymbol{x}_i$ *is the data generated by the model when given* $\boldsymbol{c}_0$*,* $\boldsymbol{x}_j$ *is the data generated by the model when given* $\boldsymbol{c}_1$*, and* $\boldsymbol{x}_k$ *is the data generated by the model when given* $\boldsymbol{c}_d$*, etc. The dataset contains data labeled as* $\boldsymbol{c}_0$*,* $\boldsymbol{c}_1$*,* $\boldsymbol{c}_d$*, etc.*

Proof. In closed-form flow matching model, the generated data is entirely from the dataset. Consider the optimal transmission path $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$, and the loss function $\sum ||\boldsymbol{u}_t(\boldsymbol{x}_t, \boldsymbol{c}) - [(1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1])||^2$. The flow field at condition $\boldsymbol{c}_0$ is:

$$\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_0) = \frac{\sum_i^m p_{t,i}\boldsymbol{e}_{t,i}}{\sum_i^m p_{t,i}} \tag{11}$$

$$\boldsymbol{e}_{t,i} = \frac{\boldsymbol{x}' - t\boldsymbol{x}_{1,i}}{1-t} \tag{12}$$

$$p_{t,i}(\boldsymbol{x}_1, \boldsymbol{x}') = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_t|^{\frac{1}{2}}} \exp[-\frac{1}{2(1-t)}||\boldsymbol{e}_{t,i}||^2] \tag{13}$$

$\boldsymbol{e}_{t,i}$ is the noise that make $\boldsymbol{x}_i$ to $\boldsymbol{x}'$ at time t. $p_{t,i}(\boldsymbol{x}_1, \boldsymbol{x}')$ is probability density of $\boldsymbol{e}_{t,i}$. $\boldsymbol{x}_i$ is data with label $\boldsymbol{c}_0$ in the dataset, $\boldsymbol{x}_j$ is data with label $\boldsymbol{c}_1$ in the dataset, etc. m is the number of data with label $\boldsymbol{c}_0$, etc, and n is the number of data with label $\boldsymbol{c}_1$ in the dataset, etc.

Thus,

$$\boldsymbol{u}_t(\boldsymbol{x}', a_0\boldsymbol{c}_0 + a_1\boldsymbol{c}_1 + ... + a_d\boldsymbol{c}_d) \tag{14}$$

$$= a_0\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_0) + a_1\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_1) + ... + a_d\boldsymbol{u}_t(\boldsymbol{x}', \boldsymbol{c}_d) \tag{15}$$

$$= a_0\frac{\sum_i^m p_{t,i}\boldsymbol{e}_{t,i}}{\sum_i^m p_{t,i}} + a_1\frac{\sum_i^n p_{t,j}\boldsymbol{e}_{t,j}}{\sum_j^n p_{t,j}} + ... + a_d\frac{\sum_k^o p_{t,k}\boldsymbol{e}_{t,k}}{\sum_k^o p_{t,k}} \tag{16}$$

$$= \frac{\sum_i^m\sum_j^n ...\sum_k^o p_{t,i}p_{t,j}...p_{t,k}(a_0\boldsymbol{e}_{t,i} + a_1\boldsymbol{e}_{t,j} + ... + a_d\boldsymbol{e}_{t,k})}{\sum_i^m\sum_j^n ...\sum_k^o p_{t,i}p_{t,j}...p_{t,k}} \tag{17}$$

$$= \frac{\sum_i^m\sum_j^n ...\sum_k^o p_{t,i}p_{t,j}...p_{t,k}\frac{(a_0+a_1+...+a_d)\boldsymbol{x}' - t(a_0\boldsymbol{x}_i + a_1\boldsymbol{x}_j + ... + a_d\boldsymbol{x}_k)}{1-t}}{\sum_i^m\sum_j^n ...\sum_k^o p_{t,i}p_{t,j}...p_{t,k}} \tag{18}$$

While the vector field directly defined on $\{\boldsymbol{x}^*|\boldsymbol{x}^* = a_0\boldsymbol{x}_i + a_1\boldsymbol{x}_j + ... + a_d\boldsymbol{x}_k\}$ is:

$$\boldsymbol{u}_t^*(\boldsymbol{x}', \boldsymbol{c}^*) = \frac{\sum_l^{nm...o} p_{t,l}\boldsymbol{e}_{t,l}}{\sum_l^{nm...o} p_{t,l}} \tag{19}$$

$$= \frac{\sum_l^{nm...o} p_{t,l}\frac{\boldsymbol{x}' - t(a_0\boldsymbol{x}_i + a_1\boldsymbol{x}_j + ... + a_d\boldsymbol{x}_k)}{1-t}}{\sum_l^{nm...o} p_{t,l}} \tag{20}$$

$\boldsymbol{u}_t^*(\boldsymbol{x}', \boldsymbol{c}^*)$ means the model is trained on the interpolation data.

Comparing $\boldsymbol{u}_t(\boldsymbol{x}', a_0\boldsymbol{c}_0 + a_1\boldsymbol{c}_1 + ... + a_d\boldsymbol{c}_d)$ and $\boldsymbol{u}^* +_t (\boldsymbol{x}', \boldsymbol{c}^*)$, we can see that although the two vector fields are not exactly the same, their final generated results are consistent. The difference lies in the different noise schedules they choose.

□

Lemma B.2. *The sample error for the piecewise-linear neural network driven conditional flow matching model is:*

$$|f(\boldsymbol{x}^*) - \boldsymbol{c}^*| \leq Kmax||\boldsymbol{c}_i - \boldsymbol{c}_j||^2 \tag{21}$$

Proof. Consider a subregion of the label space, Its vertices are $\boldsymbol{c}_0$, $\boldsymbol{c}_1$,..., $\boldsymbol{c}_d$. There exists a unique set of weight coefficients for $\boldsymbol{c}^*$ in d-dimensional space.

$$\begin{bmatrix} a_0 & \cdots & a_d \end{bmatrix} \begin{bmatrix} c_{0,1} & \cdots & c_{0,d} \\ \vdots & \ddots & \vdots \\ c_{d,1} & \cdots & c_{d,d} \end{bmatrix} = \begin{bmatrix} c_1^* & \cdots & c_d^* \end{bmatrix}$$

and we get:

$$\begin{bmatrix} a_0 & \cdots & a_d \end{bmatrix} = \begin{bmatrix} c_1^* & \cdots & c_d^* \end{bmatrix} \begin{bmatrix} c_{0,1} & \cdots & c_{0,d} \\ \vdots & \ddots & \vdots \\ c_{d,1} & \cdots & c_{d,d} \end{bmatrix}^{-1}$$

The sample generated by the model under condition $\boldsymbol{c}^*$ is $\boldsymbol{x}^*$:

$$\begin{bmatrix} x_0^* & \cdots & x_q^* \end{bmatrix} = \begin{bmatrix} a_0 & \cdots & a_d \end{bmatrix} \begin{bmatrix} x_{i,1} & \cdots & x_{i,q} \\ \vdots & \ddots & \vdots \\ x_{k,1} & \cdots & x_{k,q} \end{bmatrix}$$

By using the error bound of linear interpolation, the sample error is:

$$\begin{aligned} |f(\boldsymbol{x}^*) - \boldsymbol{c}^*| &\leq L_f|\boldsymbol{x}^* - \boldsymbol{x}^{gt}| && (22) \\ &= L_f| - \frac{1}{2}\sum_{i=0}^{d} a_i(\boldsymbol{c}_i - \boldsymbol{c})^T \boldsymbol{H}_{f^{-1}}(\boldsymbol{\xi}_i)(\boldsymbol{c}_i - \boldsymbol{c})| && (23) \\ &\leq L_f \frac{(d+1)}{2} Mmax||\boldsymbol{c}_i - \boldsymbol{c}_j||^2 && (24) \\ &= Kmax||\boldsymbol{c}_i - \boldsymbol{c}_j||^2 && (25) \end{aligned}$$

where L_f is Lipschitz constant. $f(\boldsymbol{x}) = \boldsymbol{y}$ is the real data distribution. $\boldsymbol{x}^{gt}$ is the data with label $\boldsymbol{c}^*$. f^{-1} is the inverse function of $f(x)$ defined on convex hull composed of $\boldsymbol{c}_0$, $\boldsymbol{c}_1$,..., $\boldsymbol{c}_d$. $\boldsymbol{H}_{f^{-1}}$ is the Hessian matrix of f^{-1}, and $||\boldsymbol{H}_{f^{-1}}|| \leq M$ in the convex hull. $max||\boldsymbol{c}_i - \boldsymbol{c}_j||^2$ represents the maximum distance of any two points in the convex hull.

□

C Interpolation Phenomenon on Label Dimension

Equation (3) provides the generation law of the piecewise-linear closed-form flow matching model. Specifically, interpolation in the label space results in corresponding interpolation in the data space. It is worth noting that the label dimension d is generally smaller than the data dimension, meaning that the interpolation coefficients derived from the labels induce lower-dimensional interpolation in the data space. In this section, a simple experimental demonstration of this phenomenon is presented.

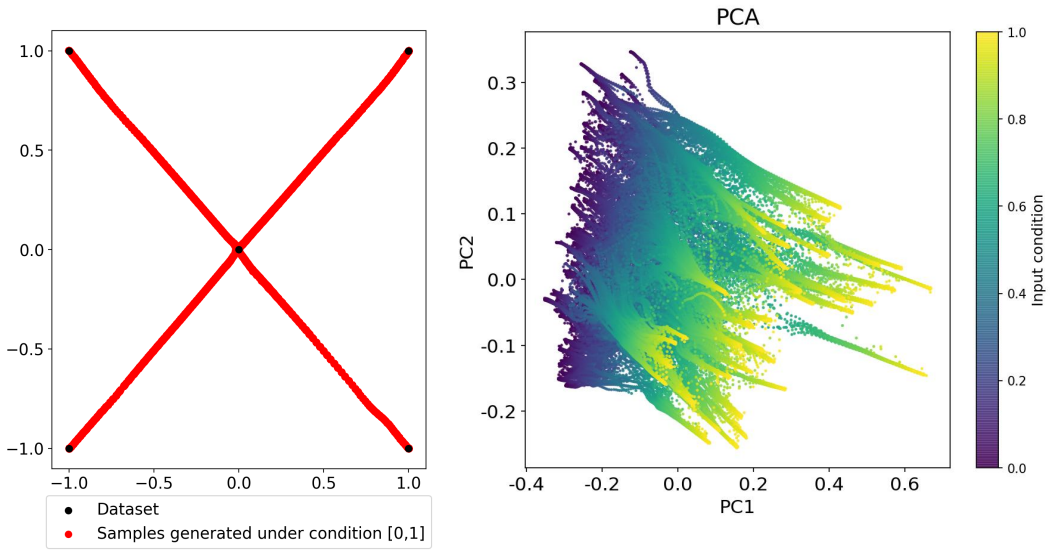


Figure 6: Interpolation phenomenon in label dimension.

Figure 6 (left) displays the training dataset and the generated results of a neural network based conditional flow matching model (Input:$\boldsymbol{y}$, Output:$\boldsymbol{x}$). The training dataset contains five data points $\boldsymbol{x} \in \{[0,0],[1,1],[-1,1],[-1,-1],[1,-1]\}$, and their labels $\boldsymbol{y} \in \{[0],[1],[1],[1],[1]\}$. Thus $\boldsymbol{x} \in \mathbb{R}^2$ and $\boldsymbol{y} \in \mathbb{R}^1$. From the figure, it can be seen that by changing the input conditions, the samples generated by the model are concentrated in the one-dimensional space of the lines connecting different labeled data in the training set. Specifically, one end of these line segments is data with label 0, and the other end is data with label 1. They form the label-dimensional interpolation ($\mathbb{R}^1$) in the data space ($\mathbb{R}^2$).

Figure 6 (right) displays the interpolation phenomenon in label dimension in Airfoil dataset (The dataset has been filtered with only 0 and 1 labels). In this complex example, the generated results are projected onto two dimensions using PCA. The results indicate that while linear variation in generated labels does not result in strictly linear variation of generated results, the samples still satisfy a continuous structural linkage between different labeled samples; specifically, interpolation of labels induces corresponding interpolation within the generated data space. Notably, Figure 6 demonstrates the rationality of piecewise-linear closed-form flow matching model theory.

As a comparison, Figure 7 shows the classification (regression) neural network (Input:$\boldsymbol{x}$, Output:$\boldsymbol{y}$) trained on the same dataset in Figure 6 (left). It can be seen that, neural network based conditional flow matching model and classification (regression) neural network display different patterns. By changing the predicted label, the data points with different prediction will fill the entire space $[-1,1] \times [-1,1]$. In addition, Figure 7 also demonstrates the piecewise-linear property of neural networks, with particular region $[-1,1] \times [-1,1]$ being divided into four sub regions, in which the neural network can be viewed as an affine transformation. The vertices of these subregions are data points in the training dataset.

This is related to the condensation phenomenon of neural network Luo et al. (2021); Xu et al. (2025). They pointed out that under certain conditions, such as when using dropout or small initialization, the parameters of neural networks may undergo condensation. This means that after fully fitting the dataset, the network tends to reduce the number of effective parameters while also decreasing the number of inflection points. As a result, the network exhibits piecewise-linear interpolation behavior in subregions, while the vertices of these subregions are data points in the training dataset.

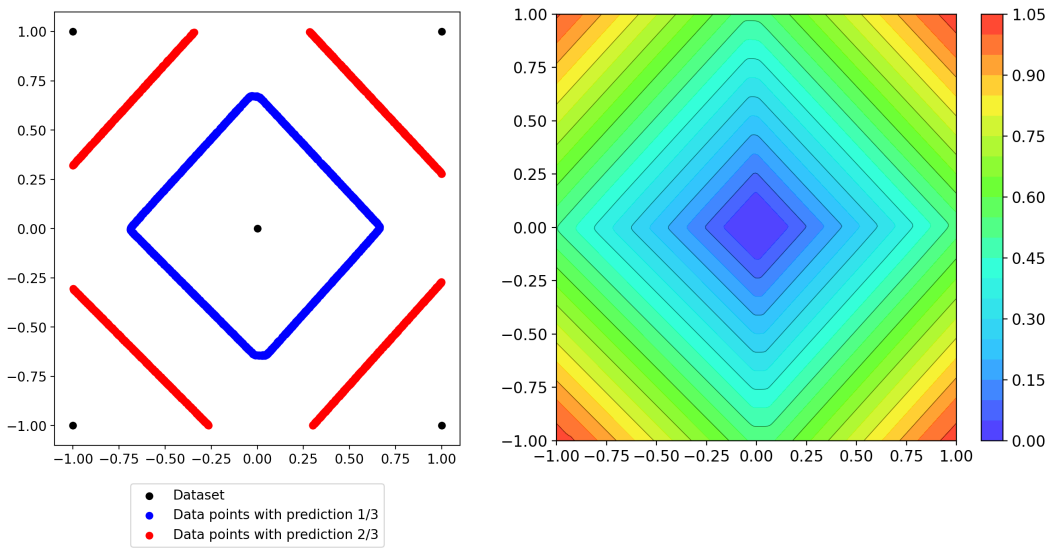


Figure 7: Predictions of classification (regression) neural network.

D SUPPLEMENTARY EXPERIMENTAL RESULTS

D.1 INTRODUCTION TO DATASETS

1) **Uneven dataset**: The first dataset is an synthetic uneven dataset, chosen for intuitive visualization of the results. Three-quarters of the data points are evenly located on the left side, while the remaining quarter are on the right. Data $\boldsymbol{x}$ are the points in $[-1, 1] \times [-1, 1]$. Their label $\boldsymbol{y} = f(\boldsymbol{x})$ is the probability density function of Gaussian distribution with $\boldsymbol{\mu}$ at $[0, 0]$.

2) **Airfoil dataset**: The second dataset is a real-world airfoil dataset from the UIUC library Selig (1996). Data $\boldsymbol{x}$ are the shape parameters of airfoils. Their labels correspond to the lift-to-drag ratio coefficients, i.e., $\boldsymbol{y} \in \mathbb{R}^1$.

3) **Flying-Wing dataset**: The third dataset is a real-world flying wing dataset. Data $\boldsymbol{x}$ are the shape parameters of flying-wing shapes. Their labels represent the working condition and the lift coefficient, namely $\boldsymbol{y} \in \mathbb{R}^3$ Wang et al. (2025).

4) **Starship dataset**: The fourth dataset is a real-world starship-like dataset Seedhouse (2022). Data $\boldsymbol{x}$ are the shape parameters of starship shapes. Their labels represent the lift coefficient, drag coefficient, pitch moment, and pressure center of the shapes, namely $\boldsymbol{y} \in \mathbb{R}^4$.

D.2 TESTING ON COMPUTING TIME

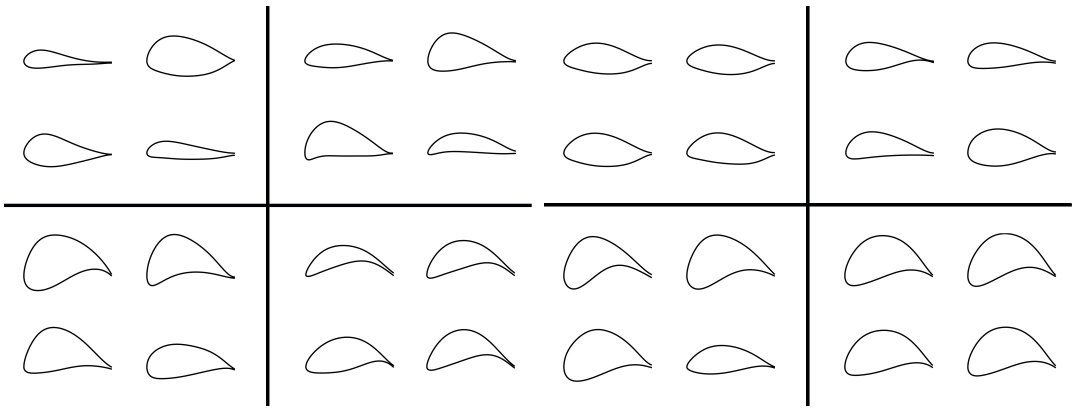
Table 1: Comparison of Query Time across Query Methods.

Method	**Time**	**Method**	**Time**	**Method**	**Time**
CoreSet	1941s	EMGM	1854s	TypiClust	1487s
Committee	572s	ProbCover	1511s	BADGE	1475s
UEADA	2152s	MaxHerding	1831s	Q_D	453s
ALIM	1978s	UHerding	2294s	Q_A	429s
ISP	6870s				

As shown in Table 1, which compares the query time of different query methods on the same dataset (Uneven) for selecting an equal percent of data points (6%). ISP is the most time-demanding, CoreSet, UEADA, ALIM, EMGM, ProbCover, MaxHerding, UHerding, TypiClust, and BADGE are the second time-demanding. This is because ISP needs to initialize a basic pre-trained generative

model. CoreSet, ALIM, ProbCover, MaxHerding, UHerding, TypiClust, and BADGE require training a neural network for feature extraction. UEADA needs to train a classification diffusion model as a proxy to estimate uncertainty. EMGM needs to train a prediction neural network as a proxy. Committe requires training of different prediction models including: SVR, Random Forest, XGBoost, and RBF neural networks. In contrast, Q_D, and Q_A exhibit similar efficiency, as their primary computational cost lies in sequentially building the RBF neural networks for each selection step.

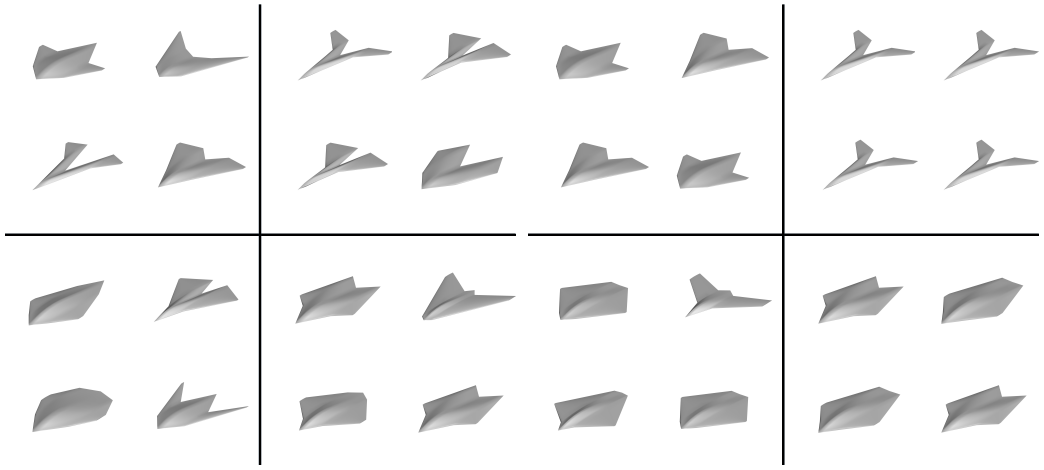
D.3 Display of generated shapes



(a) Q_D Accuracy score (MSE) :$5.73e^{-5}$ (b) Q_A Accuracy score (MSE) :$2.47e^{-5}$

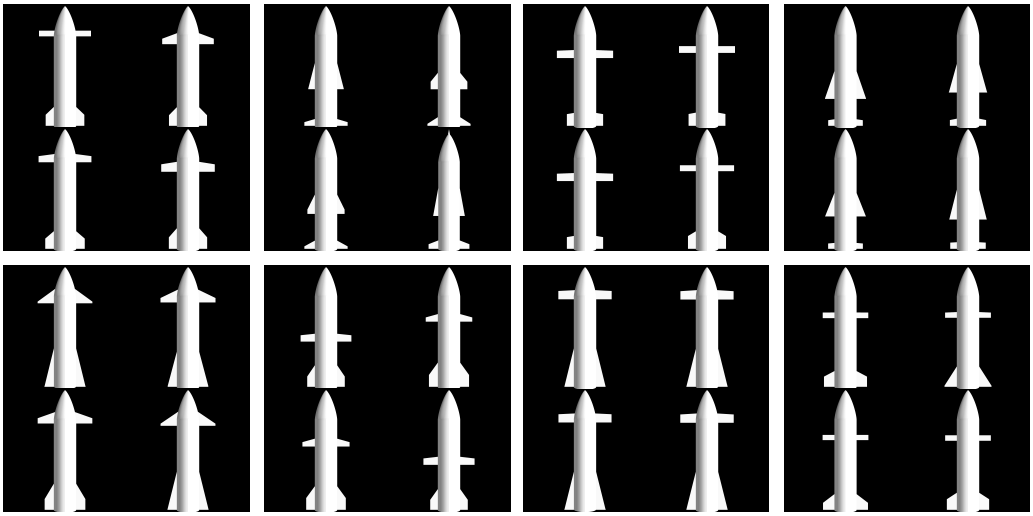
Figure 8: Comparison of generated shape results in Airfoil dataset. Generated airfoil samples under four different conditions. Each panel shows four distinct shapes corresponding to a single condition.

Figure 8, Figure 9, and Figure 10 present a comparison of samples generated by the the model trained under Q_D and Q_A query strategies across different datasets. The results demonstrate that Q_D achieves higher diversity at the cost of lower accuracy (higher MSE), whereas Q_A prioritizes accuracy, resulting in lower diversity.



(a) Q_D Accuracy score (MSE) :$5.74e^{-4}$ (b) Q_A Accuracy score (MSE) :$3.27e^{-4}$

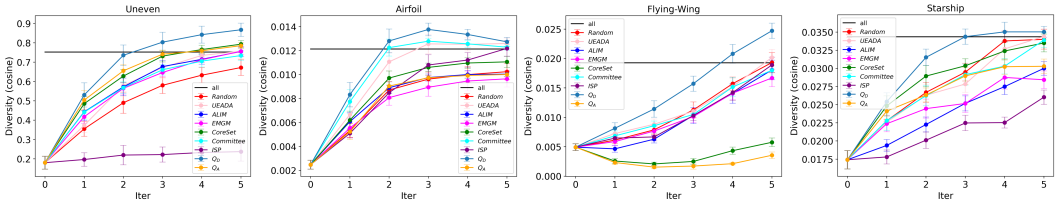
Figure 9: Comparison of generated shape results in Flying-Wing dataset. Generated flying wing samples under four different conditions. Each panel shows four distinct shapes corresponding to a single condition.



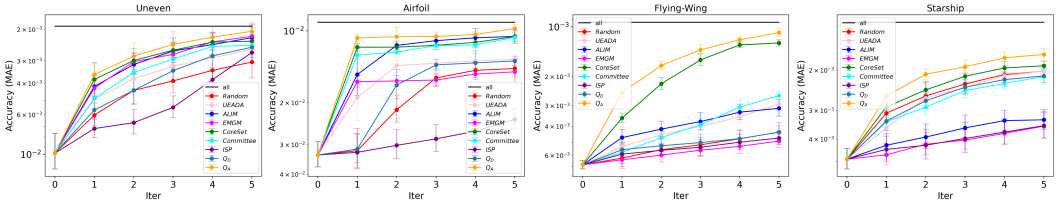
(a) Q_D Accuracy score (MSE) :$1.54e^{-3}$ (b) Q_A Accuracy score (MSE) :$1.01e^{-3}$

Figure 10: Comparison of generated shape results in Starship dataset. Generated starship samples under four different conditions. Each panel shows four distinct shapes corresponding to a single condition.

D.4 TESTING ON DIFFERENT METRICS



(a) Comparison of diversity on four datasets (higher line is better).



(b) Comparison of accuracy on four datasets (higher line is better).

Figure 11: Comparison of diversity (cosine) and accuracy (MAE) on four datasets (mean and standard deviation). For clarity, vertical coordinates of Accuracy have been reversed, making higher lines correspond to better model performance.

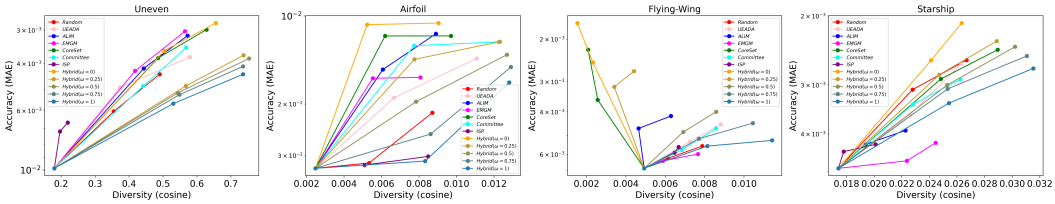
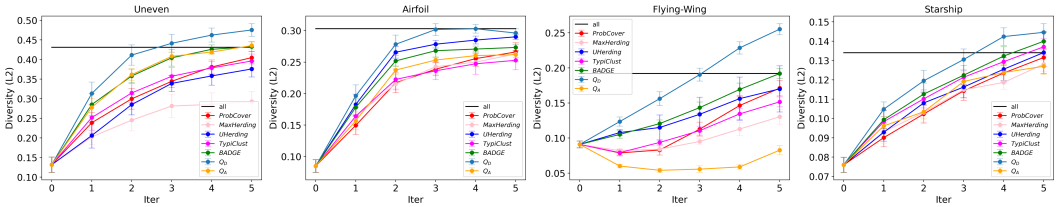


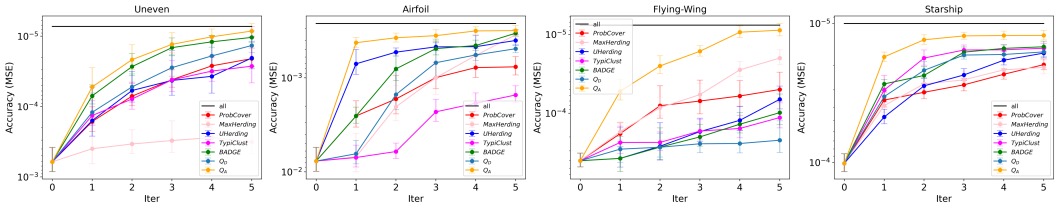
Figure 12: Comparison of diversity (cosine) and accuracy (MAE) with different ω (top right is better).

Figure 11 and Figure 12 present the results of different metrics. Specificlly, diversity score in Equation (8) is calculated with 1-similarity (scipy.spatial.distance.pdist(points, metric='cosine')) and accuracy score in Equation (9) is calculated with mean absolute error (MAE). These figures show conclusions similar to the main text.

Figure 11 compares the diversity and accuracy across the four datasets. The results indicate that Q_D achieves the highest diversity in all datasets, even outperforming the model trained on the full dataset, although this comes at the cost of reduced accuracy. In contrast, Q_A yields the highest accuracy in all datasets with the cost of reduced diversity. Figure 12 illustrates how the weight ω in Equation (7) can be tuned to control the trade-off between diversity and accuracy: a larger ω prioritizes diversity, while a smaller ω favors accuracy.

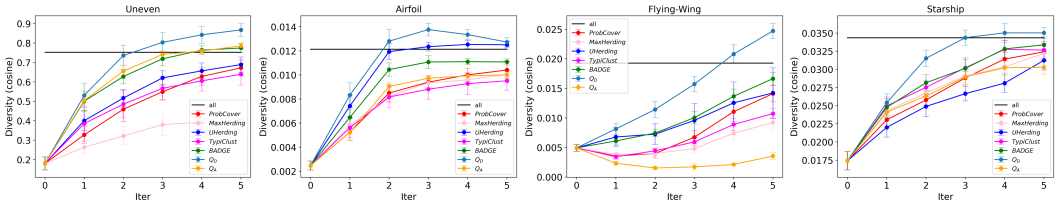


(a) Comparison of diversity on four datasets (higher line is better).

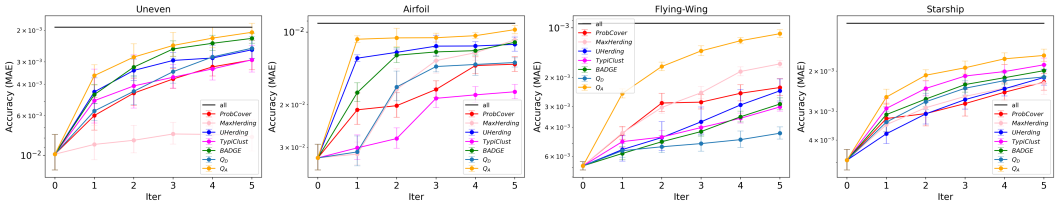


(b) Comparison of accuracy on four datasets (higher line is better).

Figure 13: Comparison of diversity (L2) and accuracy (MSE) on four datasets (mean and standard deviation). For clarity, vertical coordinates of Accuracy have been reversed, making higher lines correspond to better model performance.



(a) Comparison of diversity on four datasets (higher line is better).



(b) Comparison of accuracy on four datasets (higher line is better).

Figure 14: Comparison of diversity (cosine) and accuracy (MAE) on four datasets (mean and standard deviation). For clarity, vertical coordinates of Accuracy have been reversed, making higher lines correspond to better model performance.

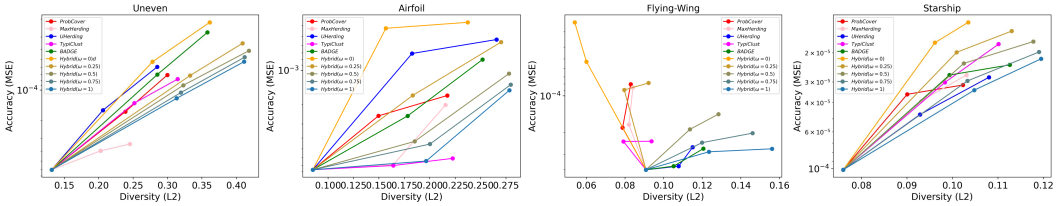


Figure 15: Comparison of diversity (L2) and accuracy (MSE) with different ω (top right is better).

D.5 More Baselines

In this section, we incorporate more advanced active learning algorithms for discriminative models, namely ProbCover Yehuda et al. (2022), MaxHerding Bae et al. (2024), UHerding Bae et al. (2025), BADGE Ash et al. (2019), and TypiClust Hacohen et al. (2022).

Figure 13 and Figure 14 compare Q_D and Q_A with these baseline using different metrics. The results demonstrate that our query strategies continue to outperform these newly added baselines.

Figure 15 and Figure 16 compare Hybrid method with these baseline using different metrics. The results demonstrate that the Hybrid method roughly constitutes a Pareto front that is not inferior to the baseline.

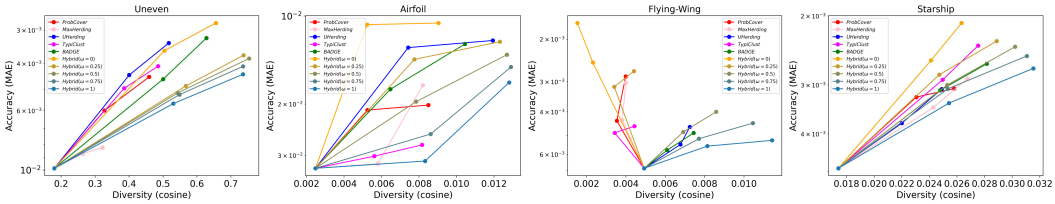


Figure 16: Comparison of diversity (cosine) and accuracy (MAE) with different ω (top right is better).

D.6 Ablation Study

The formulation of Q_D comprises three terms, the assessment of their impact of each term are presented in Figure 17 and Figure 18. The figures show that all three terms positively influence diversity. It can be seen from the figures that Case Uneven and Case Airfoil are greatly affected by $\mathrm{distance}(\mathbf{x}, \boldsymbol{\mathcal{X}})$ term, Case Flying-Wing is greatly affected by the $\mathrm{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}})$ term, and Case Starship is greatly affected by $\mathrm{distance}(\mathbf{y}, \boldsymbol{\mathcal{Y}})$ term and $\Delta\mathrm{clusterentropy}$ term.

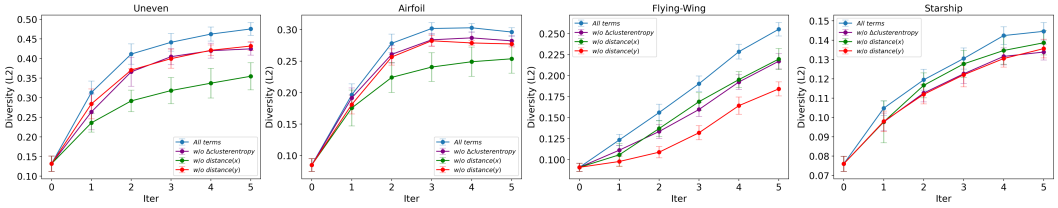


Figure 17: Ablation study on model diversity (L2) on different datasets (mean and std).

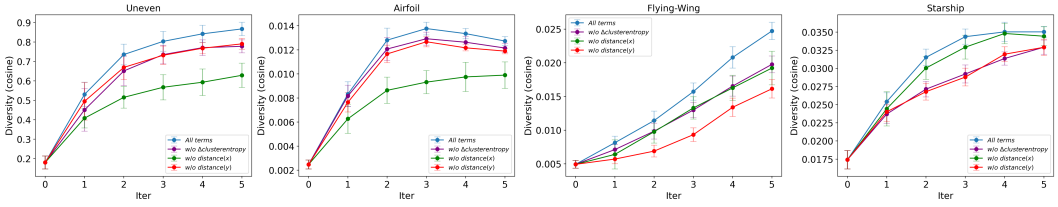


Figure 18: Ablation study on model diversity (cosine) on different datasets (mean and std).

E Hyperparameters

For our experiments, we employed a fully connected neural network with 10 layers and 512 hidden units per layer, using the LeakyReLU activation function. The model was trained with the AdamW optimizer for 4,000,000 steps with a batch size of 512. The learning rate was set to 1e-3 with a decay rate (gamma) of 0.9 applied every 100,000 steps. The model was evaluated over 100 sampling steps. For Equation (4), we set $\alpha = 1$, $\beta = 1$, and $\gamma = 1$ in each dataset as the unified hyperparameter across all datasets.

E.1 Parameter sensitivity test

To verify the robustness of Q_D under the unified hyperparameter configuration on different datasets (5 iter), we conduct a sensitivity analysis using the one-factor-at-a-time method. Specifically, we fix two hyperparameters to their unified values and vary the target parameter by 0.1. The unified configuration is highlighted in **bold**.

As shown in Table 2, the performance remains highly stable when each hyperparameter varies within a reasonable range around the unified configuration. For instance, when α varies from 0.8 to 1.2, the average performance fluctuation is small, indicating low sensitivity. The unified configuration

Table 2: Sensitivity Analysis around the Unified Hyperparameters on Diversity(L2)

Param	Value	Uneven	Airfoil	Flying-Wing	Starship
α	0.8	0.47716±0.01173	0.29775±0.00783	0.24947±0.00850	0.14587±0.00586
	0.9	0.47751±0.01733	0.29693±0.00683	0.25400±0.00953	0.14597±0.00869
	1.0	**0.47530±0.01632**	**0.29617±0.00692**	**0.25525±0.00798**	**0.14458±0.00754**
	1.1	0.47295±0.01927	0.29746±0.00591	0.26021±0.00702	0.14422±0.00865
	1.2	0.47449±0.01584	0.29887±0.00725	0.26684±0.00789	0.14449±0.00617
β	0.8	0.47449±0.01584	0.29690±0.00569	0.25345±0.00715	0.14525±0.00791
	0.9	0.47281±0.01774	0.29702±0.00570	0.25409±0.00759	0.14471±0.00716
	1.0	**0.47530±0.01632**	**0.29617±0.00692**	**0.25525±0.00798**	**0.14458±0.00754**
	1.1	0.47710±0.01588	0.29669±0.00621	0.25752±0.00708	0.14598±0.00687
	1.2	0.47481±0.01527	0.29510±0.00614	0.25601±0.00860	0.14636±0.00706
γ	0.8	0.47650±0.01503	0.29946±0.00769	0.26073±0.00802	0.14466±0.00780
	0.9	0.47638±0.02093	0.29595±0.00541	0.25818±0.00877	0.14830±0.00852
	1.0	**0.47530±0.01632**	**0.29617±0.00692**	**0.25525±0.00798**	**0.14458±0.00754**
	1.1	0.47582±0.01482	0.29629±0.00800	0.25438±0.00715	0.14624±0.00720
	1.2	0.47474±0.01286	0.29577±0.00584	0.24851±0.00790	0.14529±0.00615

Note: When varying one hyperparameter, the other two are fixed to their unified values ($\alpha = 1$, $\beta = 1$, $\gamma = 1$). The bold row in each block corresponds to the unified configuration.

(bold rows) consistently resides in the plateau region of the performance landscape across all datasets, confirming its robustness and suitability for zero-shot deployment.

E.2 Independent Tuning Results via Coordinate Descent

To further investigate the performance upper-bound and explore potential parameter interactions, we conduct independent hyperparameter tuning for each dataset using the Coordinate Descent method.

Specifically, the optimization process is initialized with the unified hyperparameter configuration ($\alpha = 1$, $\beta = 1$, $\gamma = 1$) used in the main text. In each iteration, we fix two hyperparameters and perform a 1D grid search to find the local optimum for the third.

Table 3 presents the top-5 configurations discovered by this coordinate descent process across the four datasets.

For the four datasets, optimal parameters are: Uneven: {alpha0.4:, beta:1.0, gamma:0.6}, Airfoil: {alpha:1.4, beta:1.4, gamma:0.4}, Flying-Wing: {alpha:1.8, beta: 0.2, gamma: 0.2}, Starship: {alpha: 1.0, beta: 1.6, gamma: 0.2}.

Via coordinate descent optimization, the hyperparameters independently searched for each dataset outperform the unified configuration. Specifically, the performance gains are marginal in Case Uneven, whereas they are relatively high in Case Flying-Wing.

We admit the diversity strategy has limitations, that is, the optimal parameters in different data sets is different, it is worth emphasizing that our unified hyperparameters already achieve better performance across all baselines without any dataset-specific tuning.

Table 3: Top-5 Configurations from Sequential Search across Datasets

Dataset	Rank	α	β	γ	Diversity(L2)	Gain
Uneven	1	0.4	1.0	0.6	**0.48107±0.01708**	+1.2%
	2	0.4	1.0	1.2	0.479324±0.01349	+0.8%
	3	0.4	1.0	1.4	0.47874±0.1333	+0.7%
	4	0.4	1.0	1.6	0.47861±0.01436	+0.7%
	5	0.8	1.0	01.6	0.047840±0.01409	+0.7%
Airfoil	1	1.4	1.4	0.4	**0.31674±0.01984**	+6.9%
	2	1.4	0.8	0.4	0.31503±0.01620	+6.4%
	3	1.6	0.8	0.4	0.31355±0.01503	+5.9%
	4	1.0	0.8	0.4	0.31302±0.01659	+5.7%
	5	1.2	0.8	0.4	0.31286±0.01583	+5.6%
Flying-Wing	1	1.8	0.2	0.2	**0.32887±0.01110**	+28.8%
	2	1.8	1.6	0.2	0.32603±0.01227	+27.7%
	3	1.8	0.6	0.2	0.32485±0.01509	+27.3%
	4	1.8	1.0	0.2	0.32410±0.01093	+27.0%
	5	1.8	0.8	0.2	0.32260±0.00880	+26.4%
Starship	1	1.0	1.6	0.2	**0.15529±0.00883**	+7.4%
	2	1.6	1.6	0.2	0.15339±0.01040	+6.1%
	3	1.8	1.6	0.2	0.15281±0.00739	+5.7%
	4	1.0	0.8	0.2	0.15253±0.01060	+5.5%
	5	1.0	1.0	0.2	0.15250±0.01031	+5.5%

F Statistical Validation and Analysis

To ensure the statistical reliability of our empirical findings and to rigorously evaluate the proposed method against existing baselines, we employ a comprehensive statistical validation protocol. In this protocol, we adopt non-parametric statistical tests.

F.1 Overall Performance Comparison via Friedman Test

To compare the proposed method with all 12 baseline methods simultaneously without inflating the Type I error rate (the multiple comparisons problem), we utilize the Friedman test followed by the Nemenyi post-hoc test, as recommended by Demšar (2006).

For each of the 20 random seeds, we rank the performance of all methods on the last iter. The Friedman test evaluates the null hypothesis that all methods perform equivalently. Upon rejecting the null hypothesis ($p < 0.05$), we apply the Nemenyi post-hoc test.As illustrated in Table 4, our proposed method achieves the best average rank and is statistically significantly superior to the majority of the baselines.

F.2 Rigorous Paired Testing against Strongest Baselines

While the Friedman test provides a conservative global overview, the large CD value indicates that it may lack the statistical power to distinguish between the top-performing methods due to the severe penalty of multiple comparisons.

To rigorously demonstrate the statistical superiority of our method over the most competitive baselines, we restrict our pairwise comparisons to the top two baselines. We employ the Wilcoxon signed-rank test, a non-parametric paired test robust to the outliers and skewed distributions frequently observed in early AL iterations. To ensure strict pairing, the performance of our method and the selected baselines is evaluated under identical random seeds.

Table 4: Average ranks of all methods obtained by the Friedman test across four datasets and two metrics over 20 random seeds. ↓ indicates lower is better. Rankings are computed accordingly (i.e., rank 1 corresponds to the best performance for each metric). The best average rank for each column is highlighted in **bold**.

Method	Uneven		Airfoil		Flying-Wing		Starship	
	Accuracy(MSE) (↓)	Diversity(L2) (↓)	Accuracy(MSE) (↓)	Diversity(L2) (↓)	Accuracy(MSE) (↓)	Diversity(L2) (↓)	Accuracy(MSE) (↓)	Diversity(L2) (↓)
Q_D	6.10	**1.10**	7.95	**1.50**	12.25	**1.00**	6.60	**1.65**
Q_A	**2.15**	3.50	**2.20**	10.50	**1.05**	13.95	**2.20**	10.75
Random	10.7	8.50	11.40	9.05	9.25	3.75	5.75	7.75
CoreSet	4.5	3.35	6.60	6.70	2.00	13.05	3.60	5.05
Committee	6.85	8.40	3.40	5.05	6.00	6.05	7.15	5.80
UEADA	7.05	4.65	8.50	3.20	8.10	4.70	5.25	4.95
ALIM	3.80	7.20	4.30	9.75	9.85	6.35	12.05	8.65
ISP	12.25	14.00	14.00	8.20	12.40	2.75	13.80	14.00
EMGM	4.10	8.45	10.70	11.75	12.85	4.80	13.05	11.20
ProbCover	10.15	8.40	10.90	9.00	4.85	9.70	9.50	8.60
MaxHerding	14.00	12.95	3.40	10.10	2.95	11.75	9.35	10.25
UHerding	9.30	11.20	5.35	2.45	5.85	8.80	6.95	7.450
TypiClust	10.15	9.20	13.00	11.35	9.15	10.80	4.95	5.450
BADGE	3.90	4.10	3.30	6.40	8.45	7.55	4.80	3.450

Friedman p-value: < 0.001 for all dataset-metric combinations.
Nemenyi CD ($\alpha = 0.05$)*:* [4.4369]

The results of the Wilcoxon test are summarized in Table 5. Across all four datasets and two metrics, our method is compared against the top two dataset-specific baselines. Among them, our method achieves a statistically significant improvement in all comparisons ($p < 0.05$), with effect sizes (rank-biserial correlation) ranging from 0.5238 to 1.000, indicating a practical significance of the performance gain. This confirms that while the Friedman test conservatively groups the top methods, the paired Wilcoxon test successfully identifies the significant and substantial advantage of our proposed method.

Table 5: Wilcoxon signed-rank test results comparing the proposed method against the top two strongest baselines **specific to each dataset** over 20 random seeds. Bold p-values indicate statistical significance ($p < 0.05$).

Dataset	Metric	Compared Baseline	W	p-value	Effect Size (r)
Uneven	Diversity(L2)	CoreSet	0	**1.9073e-6**	1.0000
		BADGE	1	**3.8147e-6**	0.9905
	Accuracy(MSE)	ALIM	37	**9.4357e-3**	0.6476
		BADGE	19	**5.8556e-4**	0.8190
Airfoil	Diversity(L2)	Uherding	21	**8.5068e-4**	0.8000
		UEADA	5	**1.9073e-5**	0.9524
	Accuracy(MSE)	BADGE	50	**3.9989e-2**	0.5238
		Committee	48	**3.2768e-2**	0.5429
Flying-Wing	Diversity(L2)	ISP	0	**1.9073e-6**	1.0000
		Random	0	**1.9073e-6**	1.0000
	Accuracy(MSE)	CoreSet	13	**1.6785e-4**	0.8762
		Maxherding	0	**1.9073e-6**	1.0000
Starship	Diversity(L2)	BADGE	39	**1.2079e-2**	0.6286
		UEADA	3	**9.5367e-6**	0.9714
	Accuracy(MSE)	CoreSet	43	**1.9234e-2**	1.0000
		BADGE	32	**4.8599e-3**	0.9905

G PROXY ERROR EXPERIMENT

Both Q_D and Q_A rely heavily on calculating distances in the label space $\mathbb{Y}$ (Equation (4) and Equation (6)). However, in an active learning setting, the labels of the unqueried pool are strictly unknown. In this paper, we employ an RBF neural network to predict these missing labels. However, the label predicted by RBF neural network has errors.

In this section, we will test the relationship between the proxy error and the accuracy and diversity of the model in Case Airfoil.

The setting of this experiment is that the hyperparameters of the RBF neural network include rbf__n_centers, rbf__sigma_scale, ridge__alpha. Their range and random sampling method:

Table 6: Hyperparameters

Hyperparameters	Range	Random method
rbf__n_centers	[5,1000]	randint
rbf__sigma_scale	[0.2, 15.0]	random.uniform
ridge__alpha	[1-e6,1e-1]	log uniform random

To measure the error of the proxy, we choose average proxy errors. For example, selecting 100 unlabeled sample points in one iter, the RBF neural network will be established 100 times. Predict all remaining unlabeled data each time and compare it with the true label to obtain the proxy error. The average proxy error of this iter is calculated by averaging the proxy errors over different random seeds over these 100 instances.

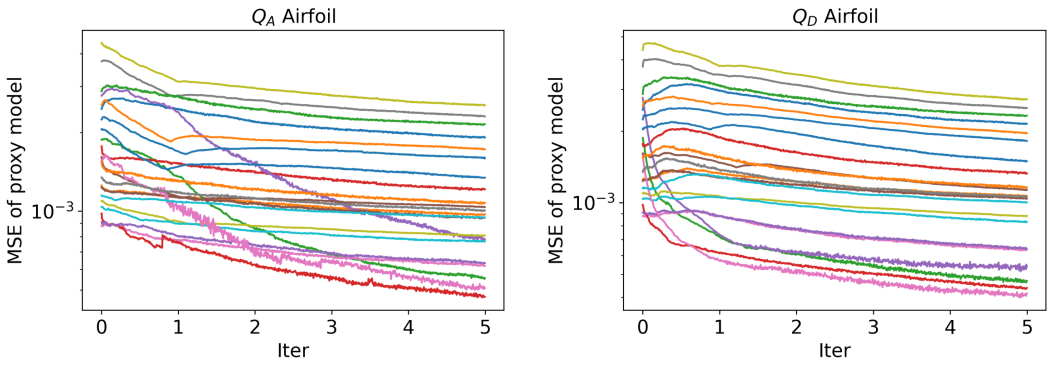


Figure 19: Proxy error of different hyperparameter proxies.

Figure 19 shows the average proxy error for different parameter proxies, and it can be seen that (1) For the same proxy, the proxy error is highest at iter1. (2) In iter3, the errors of various agents tend to stabilize.

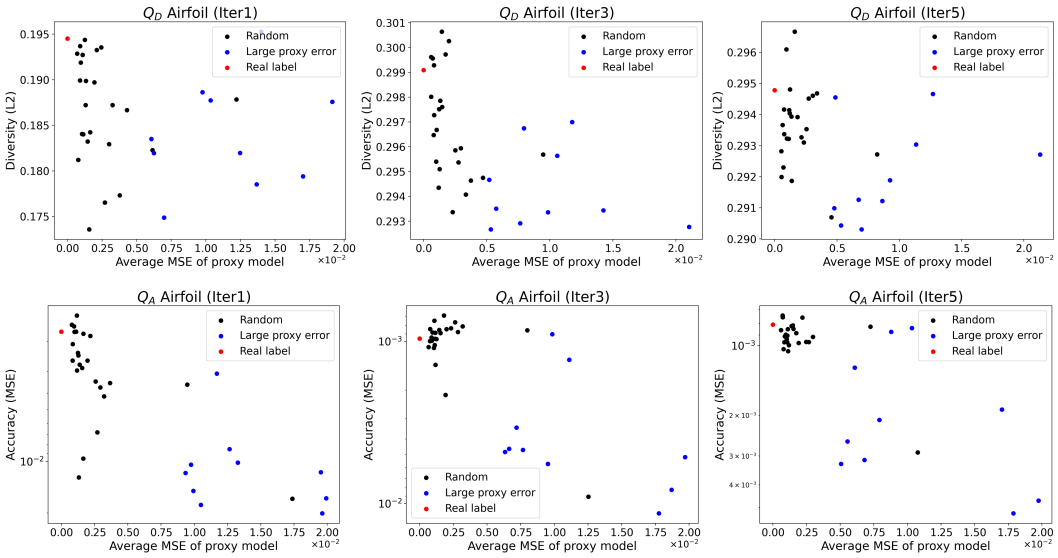


Figure 20: Comparison of different proxy errors and model metrics in different iter (A higher scatter indicates better performance).

Figure 20 shows a comparison between proxy error and model diversity and accuracy metrics. The black dots represent randomly selected hyperparameters (this represents the distribution of proxy errors when randomly selecting hyperparameters). The blue dots represent larger proxy errors in random selection (calculate the average proxy errors of iter 1 first, and train the conditional flow matching model if they are bigger), and the red dots represent the use of real labels.

From the graph, it can be seen that: (1) In iter1, the errors of each proxy are relatively large, so the impact on model performance is relatively random. (2) The larger the proxy error in later iter (iter3, iter5), the worse the performance of the model. (3) Although larger proxy errors can impair the performance of the model, their probability of occurrence is relatively small, even when randomly selecting hyperparameters (black points). (4) Using real labels has better performance than using proxies in earlier iterations, but it is not optimal in later iterations because proxies have enough training data in later iterations.

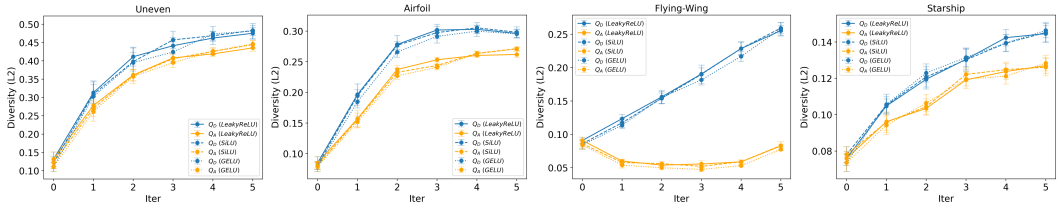
We admit that proxy errors may induce biased sample selection in the early iter. However (1) Even with random proxy hyperparameters, the model is likely to achieve better results in later iter. (2) The proposed strategy outperforms the baseline in the early iters. (3) Proxy errors are almost unavoidable in active learning, and most active learning algorithms use proxies (Committee, UEADA, ISP, EMGM, etc.). Even if proxies do not directly predict labels, they still need to extract features related to the labels (CoreSet, ALIM, etc.).

H Experiments under Assumption Violations

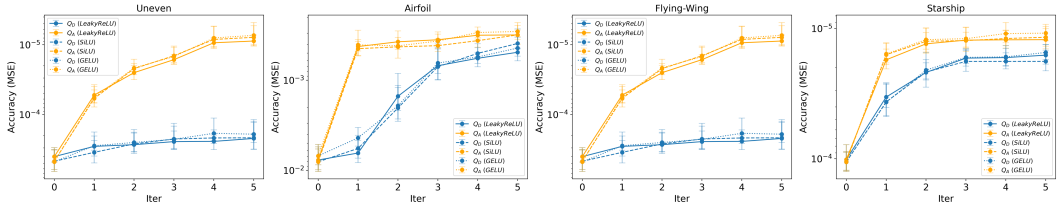
Our analytical framework relies on two assumptions: 1) the piecewise-linear assumption, and 2) the closed-form flow matching model assumption. In this section, we empirically evaluate the effectiveness of the proposed method when these assumptions are deliberately relaxed or violated.

H.1 The violation of the CPWL model assumption

The piecewise-linear assumption is grounded in the piecewise-linearity of activation functions. To satisfy the CPWL theoretical assumption, the experiments in main text utilize an MLP backbone strictly equipped with LeakyReLU activations. To relax this assumption, we present experimental results using different non-linear activation functions (SiLU and GELU) in this section.



(a) Comparison of diversity on four datasets (higher line is better).



(b) Comparison of accuracy on four datasets (higher line is better).

Figure 21: Comparison of different activation fuction among Q_D and Q_A on four datasets. It can be seen that the activation function has little impact on the performance of the model.

The experimental results of different activation functions are shown in Figure 21, which indicate that different activation functions have little effect on the performance of Q_A and Q_D. Because the data selection logic (label distribution analysis) is architecture-agnostic.

H.2 THE VIOLATION OF THE CLOSED-FORM MODEL ASSUMPTION

In general, the trained unconditional flow matching model exhibits behavior similar to the closed-form model at small t, but diverges at large t. This discrepancy arises from two main factors:

- Effective Spatial Coverage of the Vector Field: The closed-form assumption relies on minimizing the flow matching loss, which implies that the trained vector field u matches the closed-form vector field across the entire space at every time step t. However, this is invalid in practical neural network training. According to the training formulation (which transports noise $\boldsymbol{x}_0$ to data $\boldsymbol{x}_1$), when t is large, the perturbed samples used for training are concentrated near the original data manifold. Thus, the vector field is only learned locally in the vicinity of the samples. In contrast, when t is small, the noisy samples are distributed throughout the entire space, enabling the vector field to be trained globally.
- High- and Low-Frequency Errors Qian et al. (2024): At large t, the vector field is predominantly low-frequency, whereas at small t, it becomes high-frequency. Due to the inherent spectral bias of neural networks, they are naturally adept at fitting low-frequency signals but struggle to capture high-frequency components. These two fundamental differences cannot be mitigated merely by increasing the number of training epochs."

To empirically validate our analysis of the trained unconditional flow matching model, we train an unconditional flow matching model on the Uneven dataset (since the closed-form flow matching model is inherently an unconditional model) and report the vector field errors at different time steps t. Specifically, we evaluate the errors using two different protocols: 'Fixed noise' and 'Sampling trajectory'. The 'fixed noise' protocol consistently computes the vector field error across the entire space(standard Gaussion). In contrast, the 'sampling trajectory' protocol evaluates the error along the sampling trajectories of the closed-form flow matching model, that is, the error is measured near the data manifold when t is large, and across the entire space when t is small.

t	0.000	0.488	0.744	0.872	0.936	0.968	0.984	0.992	0.996	0.998	0.999
Fixed noise	4.32×10^{-5}	3.63×10^{-4}	3.36×10^{-2}	7.50×10^{-1}	9.38×10^{0}	7.47×10^{1}	4.26×10^{2}	2.03×10^{3}	8.87×10^{3}	3.71×10^{4}	1.52×10^{5}
Sampling trajectory	4.32×10^{-5}	7.23×10^{-5}	9.19×10^{-5}	1.83×10^{-4}	6.21×10^{-4}	6.68×10^{-3}	4.55×10^{-2}	2.25×10^{-1}	5.09×10^{-1}	8.80×10^{-1}	1.39×10^{1}

Table 7: Comparison of Fixed noise and Sampling trajectory over time t.

By comparing the Mean Squared Error (MSE) of the outputs between the optimal model and the practically trained model, we can empirically observe these two discrepancies. Specifically, the observation that the MSE under the fixed noise setting is greater than that along the sampling trajectory (especially as t approaches 1) validates Reason 1. Furthermore, the phenomenon that the MSE along the sampling trajectory is smaller at small t and larger at large t corroborates Reason 2. Consequently, these empirical results demonstrate that the practically trained model used in our experiments inherently deviates from the optimal model.

I POSITIONING THE PROPOSED METHOD RELATIVE TO EXPLORATION-EXPLOITATION STRATEGIES

the Exploration-Exploitation, serving as a high-level guiding principle in active learning, can be defined as:

- **Exploration**: Maximizing the coverage of the data space.
- **Exploitation**: Minimizing the uncertainty of the current model.

The concept of data space coverage is inherently model-agnostic, applying equally to both generative and discriminative models. In contrast, regarding the uncertainty of the current model, generative models cannot estimate predictive uncertainty as straightforwardly as their discriminative counterparts Zhang et al. (2024).

The challenges in conducting active learning on generative models:

1. Generative models inherently lack a well-defined notion of uncertainty.

2. The open-ended nature of querying, which differs from the closed form of querying in discriminative models that typically target a single concept. For instance, discriminative models can directly pass all samples from the unlabeled pool through the model to estimate predictive uncertainty. In contrast, generative models inherently lack the mechanism to estimate uncertainty on these unlabeled samples.

Consequently, some researchers have introduced the concept of anchor directions to reformulate the querying process as a semi-open problem[1], which estimates uncertainty only under specified conditions rather than open-ended ones. We included this method as one of the baselines in our original manuscript.

Our key insight is that, given the inherent difficulty in directly estimating the uncertainty of existing generative models, we must rethink the Exploration-Exploitation.

Consequently, we reformulate the Exploration-Exploitation as follows:

- **Exploration**: Maximizing the coverage of the data space.
- **Exploitation**: Maximizing the current model's **performance metrics**.

The rationale behind this is that, ideally in discriminative models, predictive uncertainty can be interpreted as an estimate of the model's performance metrics on the unlabeled pool. Specifically, low uncertainty implies a high likelihood of accurate predictions, whereas high uncertainty indicates a high probability of prediction errors.

However, since the querying process in practical generative models remains open-ended, we are still unable to directly estimate the model's performance metrics. We must rethink the Exploration-Exploitation again.

Consequently, we reformulate the Exploration-Exploitation as follows:

- **Exploration**: Maximizing the coverage of the data space.
- **Exploitation**:Maximizing the **ideal** model's **performance metrics**.

Given that the performance metrics of practically trained models remain difficult to estimate, we shift our focus to an idealized model whose metrics are inherently estimable. Therefore, we formulate the CPWL closed-form flow matching framework, an idealized model whose performance metrics can be directly estimated.

Derived from this analytical framework, we formulate both Q_A and Q_D, which implies that both strategies are inherently exploitation-based.

Interestingly, even though they are fundamentally exploitation-based, Q_A and Q_D also exhibit an exploratory nature: Q_A explores the label space, whereas Q_D explores the data space. This suggests that in generative models, the Exploration-Exploitation trade-off is not inherently a zero-sum game, but the two paradigms are deeply intertwined.

AI USE STATEMENT

In this paper, we first manually wrote the paper, then polished it using a large language model, and finally manually calibrated it to avoid the polished results is differ from their original meaning.