Asymptotic Optimality for Active Learning Processes Supplementary Materials

CONTENTS

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

1 PROOFS

1.1 PROOF OF THEOREM 1

Theorem 1. *(Hoeffding Inequality with IWERM) Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n_t}$ be n_t instances that are sampled from the instrumental distribution $Q(\mathbf{x}, y)$. Denote r.v. $\mathbf{S} = R(\theta) - R_t^w(\theta)$ that takes over θ, and let $b = \sup \mathbf{S}$, $a = \inf \mathbf{S}$, $E[\mathbf{S}] = \eta$. $\forall \epsilon > 0$, we have*

$$\mathbb{P}\Big(\big|R(\theta) - R_t^w(\theta)\big| \geq \epsilon\Big) \leq 2e^{\frac{-2n_t(\epsilon-\eta)^2}{(b-a)^2}}. \tag{1}$$

Proof. Firstly, assuming that $\mathbf{X}$ be real-valued random variable and $E[\mathbf{X}] = \eta$. $\forall \lambda > 0$, we have Markov's inequality:

$$\mathbb{P}\left(\mathbf{X} \geq \epsilon\right) = \mathbb{P}\left(e^{\lambda \mathbf{X}} \geq e^{\lambda \epsilon}\right) \leq e^{-\lambda \epsilon} E[e^{\lambda \mathbf{X}}]. \tag{2}$$

Following Hoeffding's lemma,

$$E[e^{\lambda \mathbf{X}}] \leq \exp\big(\lambda \eta + \tfrac{\lambda^2 (b-a)^2}{8}\big). \tag{3}$$

Considering $\mathbf{S}$ and S_i for each $(\mathbf{x}_i, y_i)$ and $a_i \leq S_i \leq b_i$. Using (2) and (3), we have

$$\mathbb{P}(\mathbf{S} \geq \epsilon) = \mathbb{P}\Big(\sum_{i=1}^{n_t} S_i \geq N\epsilon\Big) \leq \Big(\prod_{i=1}^{n_t} E_Q[e^{\lambda S_i}]\Big) e^{-n_t \lambda \epsilon} \leq \Big(\prod_{i=1}^{n_t} e^{(\lambda \eta + \frac{\lambda^2 (b_i - a_i)^2}{8})}\Big) e^{-n_t \lambda \epsilon}. \tag{4}$$

Minimizing over $\lambda \geq 0$,

$$\mathbb{P}(\mathbf{S} \geq \epsilon) \leq \min_{\lambda \geq 0} \exp\Big(\frac{n_t \lambda^2 (b-a)^2}{8} - n_t \lambda \epsilon + n_t \lambda \eta\Big) = \exp\big(-\tfrac{2n_t(\epsilon-\eta)^2}{(b-a)^2}\big). \tag{5}$$

Finally,

$$\mathbb{P}\left(|\mathbf{S}| \geq \epsilon\right) \leq 2\exp\big(-\tfrac{2n_t(\epsilon-\eta)^2}{(b-a)^2}\big). \tag{6}$$

□

RQ1: What is the difference between **Theorem 1** and [Beygelzimer et al., 2009]'s **Theorem 1**?

Ans1: Both the two theorems aim to provide a safe guarantee – consistency, but [Beygelzimer et al., 2009]'s **Theorem 1** only applies to stream-based AL, while ours applies to any AL, including stream-based AL and pool-based AL as $B \geq 1$. Specifically, in Beygelzimer et al. [2009], they make use of the martingale property and applied Azuma's inequality to get the bound, while our paper use Hoeffding's inequality to get the bound. The difference between the two inequalities is: Hoeffding proved this result for independent variables rather than martingale differences, and also observed that light modifications of his argument establish the result for martingale differences. From the perspective of AL, [Beygelzimer et al., 2009], since their theorem could only be applied to stream-based AL, where the data samples come in order, or pool-based AL with batch size as 1. They calculate the conditional expectation of $E[Z_t|Z_t - 1, ..., Z_0]$, while $Z_t = \sum(U_t, ..., U_0)$. In contrast, our theorem could be applied to any kind of AL sampling scheme, both stream-based AL and pool-based AL with $B \geq 1$.

1.2 PROOF OF PROPOSITION 1

***Proposition* 1.** *(*Asymptotic Variance of Estimators*) Let $R_t^w(\theta) = \frac{1}{\sum_{i=1}^{n_t} \beta(\mathbf{x}_i, y_i)} \sum_{i=1}^{n_t} \beta(\mathbf{x}_i, y_i) l(f(\mathbf{x}_i; \theta), y_i)$ and $R(\theta) = E_{(\mathbf{x},y)\sim P}[l(f(\mathbf{x}; \theta), y)] = \iint l(f(\mathbf{x}; \theta), y) P(\mathbf{x}, y) d\mathbf{x} dy$, by employing "Delta Method", we have*

$$\sqrt{n_t}(R_t^w(\theta) - R(\theta)) \xrightarrow{n_t \to \infty} \mathcal{N}(0, \sigma_Q^2), \tag{7}$$

with $\sigma_Q^2 = \iint \beta(\mathbf{x}, y)[l(f(\mathbf{x}; \theta), y) - R(\theta)]^2 P(\mathbf{x}, y) d\mathbf{x} dy$.

Proof. Take $l_i = l(f(\mathbf{x}_i;\theta), y_i)$, $l = \{l_1, ..., l_i, ..., l_{n_t}\}$ $\beta_i = \beta(\mathbf{x}_i, y_i)$, $\beta = \{\beta_1, ..., \beta_i, ..., \beta_{n_t}\}$, $r_t = \sum_{i=1}^{n_t} \beta_i l_i$, $R_t = R_t^w(\theta) = \frac{1}{n_t} r_t$, $R = R(\theta)$ and $B_{n_t} = \sum_{i=1}^{n_t} \beta_i$, $\mathbf{l}_{n_t} = \frac{1}{n_t} \sum_{i=1}^{n_t} l_i$.

Since the data samples are drawn from Q distribution, we have $E_Q[R_t] = R$, $E_Q[r_t] = n_t R$, and $E_Q[B_{n_t}] = n_t$. The random variables $\beta_1, ..., \beta_{n_t}$ and $\beta_1 l_1, ..., \beta_{n_t} l_{n_t}$ are $i.i.d.$, by using CLT, we have

$$\sqrt{n_t}(\frac{1}{n_t} r_t - R) \overset{n_t \to \infty}{\longrightarrow} \mathcal{N}(0, Var(\beta l)) \tag{8}$$

$$\sqrt{n_t}(\frac{1}{n_t} B_{n_t} - 1) \overset{n_t \to \infty}{\longrightarrow} \mathcal{N}(0, Var(\beta)) \tag{9}$$

Assuming $g(u, v) = \frac{u}{v}$, let $u = \frac{1}{n_t} r_t$, $v = \frac{1}{n_t} B_{n_t}$. We then use multivariate delta method to get

$$\sqrt{n_t}(g(u, v) - g(E[u], E[v]) = \sqrt{n_t}(\frac{1}{B_{n_t}} r_t - \frac{R}{1}) \overset{n_t \to \infty}{\longrightarrow} \mathcal{N}(0, \nabla g^T \Sigma \nabla g) \tag{10}$$

where $\nabla g = \nabla(\frac{r_t}{B_{n_t}}) = \nabla g(R, 1)$ represents the gradient of g and Σ is the covariance matrix of r_t and B_{n_t}.

$$\Sigma = \begin{pmatrix} Var(\beta l) & Cov(\beta l, \beta) \\ Cov(\beta, \beta l) & Var(\beta) \end{pmatrix}$$

Then we calculate

$$\begin{aligned} \nabla g(R, 1)^T \Sigma \nabla g(R, 1) &= Var(\beta l) - 2RCov(\beta l, \beta) + R^2 Var(\beta) \\ &= E[(\beta l)^2] - E[(\beta l)]^2 - 2R(E[\beta^2 l] - R * 1]) + R^2(E[\beta^2] - E[\beta]^2) \\ &= E[(\beta l)^2] - R^2 - 2RE[\beta^2 l] + 2R^2 + R^2(E[\beta^2] - R^2 * 1) \\ &= E[(\beta l)^2] - 2RE[\beta^2 l] + R^2 E[\beta^2] \\ &= \iint \beta(\mathbf{x}, y)^2 (l_i - R)^2 Q(\mathbf{x}, y) d\mathbf{x} dy \\ &= \iint \beta(\mathbf{x}, y)[l(f(\mathbf{x};\theta_t), y) - R(\theta)]^2 P(\mathbf{x}, y) d\mathbf{x} dy \end{aligned} \tag{11}$$

□

Note that in the proof of **Proposition 1**, we utilize original form $R_t^w(\theta) = \frac{1}{\sum_{i=1}^{n_t} \beta(\mathbf{x}_i, y_i)} \sum_{i=1}^{n_t} \beta(\mathbf{x}_i, y_i) l(f(\mathbf{x}_i;\theta), y_i)$. In our full paper, to facilitate the calculation, we use $R_t^w(\theta) = \frac{1}{n_t} \sum_{i=1}^{n_t} \beta(\mathbf{x}_i, y_i) l(f(\mathbf{x}_i;\theta), y_i)$, while $\lim_{n_t \to \infty} \beta(\mathbf{x}, y) = 1$ and $E_Q[\beta(\mathbf{x}, y)] = 1$.

1.3 PROOF OF PROPOSITION 2

***Proposition* 2.** *(*Optimal Sampling Distribution*) The optimal instrumental sampling distribution that minimizes σ_Q^2 is*

$$Q_t^{opt}(\mathbf{x}, y) \propto \big|l(f(\mathbf{x};\theta), y) - R(\theta)\big| P(\mathbf{x}, y). \tag{12}$$

Proof. We minimize the variance estimate σ_Q^2 in terms of Q under the constraint $\iint Q(\mathbf{x}, y) d\mathbf{x} dy = 1$ using Lagrange multiplier τ.

$$\mathcal{L}[Q, \tau] = \sigma_Q^2 + \tau(\iint Q(\mathbf{x}, y) d\mathbf{x} dy - 1) \tag{13}$$

$$= \iint \frac{\Lambda(\mathbf{x}, y)}{Q(\mathbf{x}, y)} + \tau(Q(\mathbf{x}, y) - 1) d\mathbf{x} dy, \tag{14}$$

where $\Lambda(\mathbf{x}, y) = P(\mathbf{x}, y)^2[l(f(\mathbf{x};\theta), y) - R(\theta)]^2$.

We define $G(Q(\mathbf{x}, y); \mathbf{x}, y) = \frac{\Lambda(\mathbf{x},y)}{Q(\mathbf{x},y)} + \tau(Q(\mathbf{x}, y) - 1)$. The optimal point for the constrained problem satisfies the Euler-Lagrange equation:

$$\frac{\partial G}{\partial Q(\mathbf{x}, y)} = -\frac{\Lambda(\mathbf{x}, y)}{Q(\mathbf{x}, y)^2} + \tau = 0. \tag{15}$$

A solution $w.r.t$ the normalization constraint is:

$$Q^* = \frac{\sqrt{\Lambda(\mathbf{x}, y)}}{\iint \sqrt{\Lambda(\mathbf{x}, y)} d\mathbf{x} dy}. \tag{16}$$

Since Q is a sampling distribution, we dismiss the negative solution. Substituting Λ into (16), we have

$$Q^*(\mathbf{x}, y) \propto \big|l(f(\mathbf{x};\theta), y) - R(\theta)\big| P(\mathbf{x}, y). \tag{17}$$

□

These proofs are with reference of [Sawade et al., 2010].

2 ADDITIONAL RELATED WORK

This Section is the supplement of the Section Related Work in full paper, which mainly discusses the difference between [Farquhar et al., 2021] and our work. Both [Farquhar et al., 2021] and our work focus on the bias problems resulting from the AL processes. In [Farquhar et al., 2021], they construct unbiased estimator of empirical risk R_{Labelled} by R_{Pool} with weighted loss. They optimize the intended objective, not for minimize the train – test gap. Different from [Farquhar et al., 2021], we construct (asymptotic) unbiased estimator of the expectation of loss $E_{\mathcal{X}\times\mathcal{Y}\sim P}[\text{loss}]$ by IWERM ($\beta R_{\text{Labelled}}$). In [Farquhar et al., 2021], their assumptions of constructing unbiased estimator during AL processes are 1) data that are sampled uniformly from $\mathcal{D}_{\text{Pool}}$ is unbiased (with probability $\frac{1}{N}$ if $|\mathcal{D}_{\text{Pool}}| = N$) and 2) the selection probability must be non-zero on all of the training data. In our work, we construct the (asymptotic) unbiased estimator during the AL processes by the assumption: data in $\mathcal{D}_{\text{Pool}}$ are sampled $i.i.d.$ from underlying distribution $P(\mathbf{x}, y)$ and each data $(\mathbf{x}_i, y_i)$ are sampled with probability $P(\mathbf{x}_i, y_i)$. In [Farquhar et al., 2021], they construct the acquisition proposal distribution from the perspective of the risk estimation itself. In our work, we construct the acquisition proposal distribution from the existing AL strategies. To sum up, [Farquhar et al., 2021] aims to "remove the bias" during AL training processes, while our work aims to model the difference between the underlying distribution of the whole data space and the sampling distribution generated by AL strategies.

3 SUPPLEMENT OF METHODOLOGY

3.1 INDEPENDENCE ASSUMPTION

In main paper, we discussed the independence assumption in Section 1. The whole AL process is changing constantly with the labeled set and basic model updating in each stage, and thus it is not enough to just collect data "actively" and treat the model fitting part the same as passive learning. For passive learning, one key assumption is that the training set comprises *i.i.d.*samples from the unknown true data distribution $P(\mathbf{x}, y)$, $\mathcal{D}_n \overset{i.i.d.}{\sim} P$.

If we select data samples sequentially by some fixed heuristics in AL (e.g., uncertainty-based strategies), the labeled training set is **not** drawn *i.i.d.* from P. In AL sampling processes, data are sampled in different stages are not independent to each other, since the sampling strategy at stage t depends on stage $t-1$, would mixing them into $\mathcal{D}_l$ violates the needs of independence to prove decent statistical bounds? The answer is **No**. From the aspect of sample size tends to infinity, Although AL processes are not independent, after observed enough data, for both stream-based and pool-based AL, the data (including both labeled and unlabeled) are sampled *i.i.d.*from underlying data distribution (which is consistent with Section 3 in [Beygelzimer et al., 2009]). Therefore, we can still obtain the statistical bounds with independence assumption as sample size tends to infinity. From the aspect of per stage in AL sampling processes, although the current data distribution of labeled set is non *i.i.d.*, but the estimator provided by IWERM is still be unbiased, therefore, we can still obtain the statistical bounds of our learned hypothesis consistent with the statistical bounds under the independent assumptions.

3.2 DISCUSSIONS OF "NON-INFORMATIVENESS"

We analyse the representation of β_t when the sample size tends to infinity. Suppose that $R_t^w(\theta)$ is an unbiased estimator of $R(\theta)$, which is based on a sufficiently strong classifier (e.g., using CNN as a basic classifier). If an infinite number of samples are observed, the basic classifier will have a very certain prediction given $\mathbf{x}_i$. Thus, entropy-based uncertainty sampling will converge to "non-informativeness" as the sample size tends to infinity, since all predictions are certain.

We explain why some AL sampling strategies can converge to "non-informativeness" based on the assumptions in AL sampling processes (see Section 3.1 after Lemma 1) from 2 aspects, using entropy-based uncertainty sampling as example. We review the assumptions here: In this paper, we assume that AL would not query non-existing or out-of-distribution (OOD) data samples and would not query wrong/noisy labels from oracles/experts, that is, $P(\mathbf{x}, y) > 0$ and $Q(\mathbf{x}, y) > 0$. Additionally, we could also obtain another vital information from these assumptions: $P(y = y_{\text{true}}|\mathbf{x}_i) = 1$ for all labeled samples. Firstly, after querying enough samples, any $\mathbf{x}_i$ actually appears in the labeled trained set, and thus we know it's hard label and are very certain about it, i.e., $P(y = y_{\text{true}}|\mathbf{x}_i) = 1$, thus the confidence is 1. Secondly, [de Cossio and de Cossio Diaz, 2015] shows that the practice of using sample average as surrogates of probability expectations is reliable provided sample size is large. Equation (1) in [de Cossio and de Cossio Diaz, 2015] shows that the entropy of model parameters will converge to a certain value as sample size increases. That is, after observing enough data, any given $\mathbf{x}_i$ will not change the basic model, and thus any $\mathbf{x}_i$ is meaningfulness to improve the basic model, which is consistent with our proposed "non-informativeness" assumption.

Besides for uncertainty-based AL methods like entropy-based uncertainty sampling (**US**), some representative/diversity based AL strategies also converge to "non-informativeness". For instance, Wu et al. [2006] provided a diversity-based method, which encourages the selection of unlabeled samples that are far from the labeled set and removes the redundancy within the selected samples. The redundancy of samples is measured by the angles between the samples:

$$\text{diversity}(\mathbf{x}_i) = 1 - \max_{\mathbf{x}_j \in \mathcal{D}_l} \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) K(\mathbf{x}_j, \mathbf{x}_j)}} \tag{18}$$

where K is Mercer kernel operator and $\mathcal{D}_l$ refers to the labeled set. when the size of $\mathcal{D}_l$. when the size of $\mathcal{D}_l$ tends to infinity, then diversity$(\mathbf{x}_i)$ converges to a constant. Moreover, some combined AL strategies also converge to "non-informativeness". For instance, combining uncertainty-based and representative/diversity based methods with weighted sum optimization, if each of the components converges to a constant, the combined strategies will converge to "non-informativeness".

Next we show an example that the acquisition function of existing AL methodology would not converge to "non-informativeness". Wu et al. [2006] further provided a representativeness-based sampling scheme, which indicates that the examples with high representativeness will add more information to the training set. The representativeness of an instance can be evaluated on how many instances are similar to it. Given unlabeled data pool $\mathcal{D}_u$, $|\mathcal{D}_u| = n$, the representativeness score is defined as the average similarity of all other data in $\mathcal{D}_u$:

$$\text{representativeness}(\mathbf{x}_i) = \frac{\sum_{i \neq j} K(\mathbf{x}_i, \mathbf{x}_j)}{n - 1}. \tag{19}$$

The output of this acquisition function would not be constant as the sample size increases, since it just depends on unlabeled data pool.

It's not easy to provide a very clear and recognizable paradigm for ensuring whether an AL sampling scheme converges to "non-informativeness" or not. We should observe its acquisition function to determine whether it converges to "non-informativeness" or not. In general, most non-agnostic AL sampling schemes that make selection singly depend on the basic learned models would converge to "non-informativeness", since they are aiming to detect the disagreement of predictions of given learned models or the uncertainty of the output label, after observing enough data and obtaining well-training basic learned models, the discrepancy among unlabeled data samples could not be accessed. In contrast, most agnostic AL sampling methods like [Sener and Savarese, 2017] that make selection singly depend on the information extracted from unlabeled pool would not converge to "non-informativeness", since the unlabeled data pool is constantly changes, and the information extracted from unlabeled data pool might also be changed in each stage of AL processes.

3.3 DISCUSSIONS OF P DISTRIBUTION

How to estimate P_t distribution in practice is one key point in experimental settings. In classical ML experiments, the feature is fixed, thus we employ the fixed feature provided by the data set. In deep learning tasks, we employ the penultimate layer

Table 1: Datasets used in the experiments. The Imbalance Ratio (IR) is the ratio of the number of samples in the majority class to that of the minority class.

Dataset	# of classes	# of feature dimension	# of initial labelled set	# of unlabeled pool	# of test set	# of Maximum Budget	Imbalance Ratio
EX8a	2	2	20	325	518	325	1.0
GCloudub	2	2	20	380	600	380	2.0
R15	15	2	40	200	360	200	1.0
D31	31	2	80	1,120	1,800	1,120	1.0
Clean1	2	168	20	170	285	170	1.3
Tic-tac-toe	2	9	20	363	575	363	6.8
Splice	2	61	20	380	600	380	1.1
Vehicle	4	18	20	318	508	318	1.1

of the neural network as feature, therefore the feature changes dynamically as the updating of basic classifier. In classical ML tasks, we model $P_t(\mathbf{x}, y) = P_t(\mathbf{x}|y)P_t(y)$ using a class-conditional generative model and prior distribution of class labels. In deep learning tasks, we model $P_t(\mathbf{x}, y) = P_t(y|\mathbf{x})P_t(\mathbf{x})$ using the classifiers posterior and input data distribution. The classifier posterior $P_t(y|\mathbf{x})$ is learned from the labeled data, while input data distribution $P_t(\mathbf{x})$ is estimated from feature $\mathbf{x}$.

4 ADDITIONAL EXPERIMENTS

4.1 DATASET DESCRIPTION

See Table 1 for the details of datasets applied in our experiments, including the number of classes, the number of feature dimension, the size of initial labeled pool, the size of initial unlabeled data pool, the size of testing set and the imbalance ratio of each dataset.

4.2 BASELINES

This section shows the detail description of baseline AL models.

- **US** [Lewis and Catlett, 1994]: This method is introduced in full paper.
- **QBC**: **Query-by-Committee (QBC)** uses a committee of models $\mathcal{C} = \{\theta^{(1)}, ..., \theta^{(C)}\}$ (constructed by ensemble methods or various basic classifiers), which are trained on $\mathcal{D}_l$ to predict the labels of $\mathcal{D}_u$, and the ones with largest disagreement are selected for labeling by an oracle Seung et al. [1992], Settles [2009]. The disagreement level could be measured by Voting Entropy (VE) or KL divergence. The optimization function is:

$$x^*_{VE} = \arg\max_x -\sum_i \frac{V(y_i)}{C} \log \frac{V(y_i)}{C}, \tag{20}$$

  where $V(\cdot)$ is the voting entropy across the committee of classifiers.
- **EER**: **Expected Error Reduction (EER)** maximizes the decrease of loss by adding new data samples Roy and McCallum [2001], Settles [2009]. The optimization function is:

$$x^*_{EER} = \arg\min_x \sum_i p_\theta(y_i|x)(-\sum_{u=1}^{U}\sum_j p_{\theta^+}(y_j|x^{(u)}) \log p_{\theta^+}(y_j|x^{(u)})), \tag{21}$$

  where θ^+ refers to the newly trained model after adding new data tuple.
- **BMDR**: **Batch-mode Discriminative and Representative AL (BMDR)** Wang and Ye [2015] queries a batch of informative and representative examples by minimizing the empirical risk bound of AL.

$$\min_{\mathcal{D}_q, f} \sum\nolimits_{\{\mathbf{x},y\}\in\mathcal{D}_l} l(f, \mathbf{x}, y) + \sum\nolimits_{\mathbf{x}_i\in\mathcal{D}_q} l(f, \mathbf{x}, \hat{y}) + \lambda||f||^2 + \beta MMD(\mathcal{D}, \mathcal{D}_l \cup \mathcal{D}_q), \tag{22}$$

  where $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$. MMD is maximum mean discrepancy, is a distance on the space of probability measures which has found numerous applications in machine learning and nonparametric testing.
- **US-D**: It is **US** with Dropout regularization.

- **UPAL** [Ganti and Gray, 2012]: works by minimizing the unbiased estimator of the risk of a hypothesis in a given hypothesis space. In this work, they calculate the importance weight of data $(\mathbf{x}_i, y_i)$ as $\frac{Q_i^t}{p_i^t}$, where $Q_i^t \in \{0, 1\}$, represents whether this data sample are queried or not, and p_i^t is sampling distribution, calculated by $p_i^t = p_{\text{min}}^t + (1 - \frac{1}{np_{\text{min}}^t}) \frac{H(Pr[+1|\mathbf{x}_i, h_{\mathcal{A},t-1}])}{\sum_j H(Pr[+1|\mathbf{x}_j, h_{\mathcal{A},t-1}])}$. $H(\cdot)$ is entropy and $p_{\text{min}}^t = \frac{1}{n_t}$.
- **SWAL** [Imberg et al., 2020]: this work shows that optimal predictive performance is achieved by over-sampling influential instances and high-leverage data points, and that uncertain instances not necessarily are informative ones. **SWAL** computes sampling probabilities $p_{t,i} \in (0, 1)$ for each data point, and they update the sampling weights (w) by $w_{t,i} = w_{t-1,i} + (\frac{1}{p_{t,i} - w_{t-1,i}})$. and update the model parameter by:

$$\hat{\theta}_t = \arg\min_{\theta} \sum_i w_{t,i} l_i(f(\mathbf{x}_i; \theta), y_i). \tag{23}$$

  We employed 3 variants in this paper:

  1. **SWAL-cora** calculates the sampling probabilities by: $p_{t,i} \propto \sqrt{h_{ii}(\theta)}$, where $h_{ii}(\theta) = \text{Var}_\theta(Y_i^*|\mathbf{x}_i) x_i^T \mathbf{H}^{-1} x_i$, $\mathbf{H} = \mathbf{H}(\theta) \propto \mathbf{X}^T \mathbf{V} \mathbf{X}$ and $\mathbf{V} = \mathbf{V}(\theta)$ be the diagonal matrix of $\text{Var}_\theta(Y_i^*|\mathbf{x}_i)$.
  2. **SWAL-corb** calculates the sampling probabilities by: $p_{t,i} \propto ||\sqrt{\text{Var}_\theta(Y_i^*)} \mathbf{V} \mathbf{X} \mathbf{H}^{-1} x_i||$.
  3. **SWAL-prop** calculates the sampling probabilities by: $p_{t,i} \propto \sqrt{E_\theta[l_i(f(\mathbf{x}_i; \theta), Y_i^*)^2]}$.

US, **US-D**, **QBC**, **EER** and **BMDR** are all converge to "non-informativeness" as the sample size increases.

4.3 IMPLEMENTATION DETAILS

- In classical ML tasks, for basic classifier in various AL methods and our framework, we employed Support Vector Machine (SVM) with probability measure[1]. Note that there are some experiments were missing in presented experimental results, e.g., EER on *D31* dataset could not be completed, since the basic classifier would encounter "All samples with positive weights have the same label." error when facing some subsets. So we didn't report these performance in our experiments since they are not completed 10 repeated trials. There are 3 requirements for choosing basic learned model: 1) the basic learned model is asymptotically unbiased and consistent as sample size increases in passive learning tasks; 2) the basic learned model could output the predicted class probabilities; 3) the basic learned model could change sample weights during training.
- For the basic parameter settings of basic AL models, we followed the settings provided by ALiPy project Tang et al. [2019][2]. The experiments are based on sklearn.
- In **QBC**, we employed "Bagging meta-estimator" strategy to achieve "committee of classifiers" and the basic classifier in Bagging is the default setting in sklearn library, that is, Decision Tree classifier.
- We utilize Gaussian Naive Bayes[3] to estimate P_t in AL with classical ML tasks.
- We utilize predicted class probabilities $(P_t(y|\mathbf{x}))$ provided by basic classifiers and Kernel Density Estimator[4] (KDE) to calculate $(P_t(\mathbf{x}))$ to estimate P_t in AL with deep learning tasks.
- When splitting datasets of classical ML tasks, we have more data samples in the testing (60%) sets than the training sets (40%), since we want to observe the generalization of the basic classifiers generated by various AL methods.
- We randomly select the initial data pool $\mathcal{D}_l^0$ from the training set and the remaining un-selected training set becomes our unlabeled data pool.
- In classical ML tasks, we fixed the random seed (4666) when splitting the initial labeled/training/testing sets to ensure that considering the 10 repeated experiments, we have the same data splitting for running each AL method on each dataset.
- To avoid bias problems, we have avoided any specific dataset tuning or pre-processing.

[1] https://scikit-learn.org/0.24/modules/generated/sklearn.svm.SVC.html

[2] https://github.com/NUAA-AL/ALiPy

[3] https://scikit-learn.org/0.24/modules/naive_bayes.html

[4] https://scikit-learn.org/0.24/modules/generated/sklearn.neighbors.KernelDensity.html

- In our practical implementation, we time a coefficient on the importance-weight: $\beta_t(\mathbf{x}_i, y_i) = \frac{|\mathcal{D}_u+\mathcal{D}_l|}{|\mathcal{D}_u|}\frac{P_t(\mathbf{x}_i,y_i)}{Q_t(\mathbf{x}_i,y_i)}$, where $|\mathcal{D}_u + \mathcal{D}_l|$ refers to the full dataset size and $|\mathcal{D}_u|$ refers to the size of unlabeled data pool at stage t. This is because during estimation at each stage, P_t is estimated and normalized over the whole dataset. For Q_t – specially for calculating the importance weight of training data, it is estimated and normalized over the labelled set,. So when we calculate the importance weight β_t for training data, we have an extra normalization coefficient $\frac{|\mathcal{D}_u+\mathcal{D}_l|}{|\mathcal{D}_u|}$. To eliminate the impact of the coefficient, we should time this coefficient when calculating the importance weight for re-training the basic classifier. Note that it only works for pool-based AL.

4.4 EXPERIMENTAL RESULTS

4.4.1 Sensitive analysis

We use GaussianNB for P, which has 2 parameters: priors and variance smoothing. The priors refer to the prior probabilities of the classes. The variance smoothing refers to the portion of the largest variance of all features that is added to variances for calculation stability. can see `https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html` for more details. In the paper, we use the default settings (variance smoothing: 10^{-9}, prior: None). We conduct new experiments varying the hyperparameters using **US** on *EX8a* under $B = 10$ (see Table 2). Our model is not sensitive to the hyperparameter settings.

Table 2: Sensitive Analysis.

setting	AUBC (acc)
variance smoothing 10^{-9}	$0.839(0.013)$
variance smoothing 10^{-7}	$0.849(0.016)$
variance smoothing 10^{-5}	$0.846(0.015)$
variance smoothing 10^{-3}	$0.844(0.011)$
variance smoothing 10^{-2}	$0.843(0.013)$
variance smoothing 10^{-1}	$0.847(0.022)$
prior None	$0.839(0.013)$
prior Uniform	$0.842(0.018)$
prior Class Ratio	$0.843(0.010)$

4.4.2 Additional Experiments on Classical ML tasks

We present the accuracy vs. Budget curves with batch size settings $B = 1$, $B = 5$ and $B = 20$ (see Figures 10, 7 and 4), the AUC vs. Budget curves with batch size setting $B = 1$, $B = 5$, $B = 10$ and $B = 20$ (see Figures 11, 8, 2 and 5) and the F_1 vs. Budget curves with batch size settings $B = 1$, $B = 5$, $B = 10$ and $B = 20$ (see Figures 12, 9, 3 and 6).

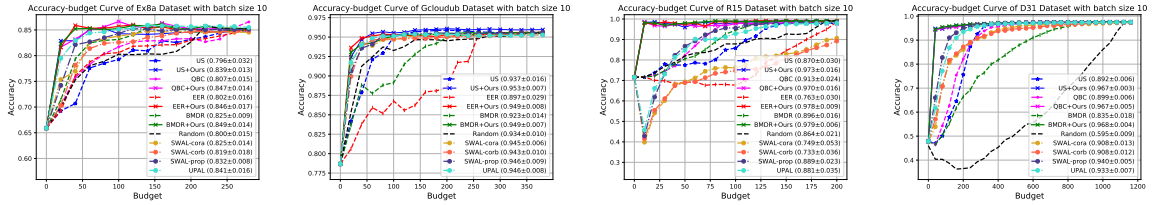
We could observe from the same with different batch size settings (i.e., $B \in \{1, 5, 10, 20\}$), our approach still improves the performance of the baseline AL model much and maintains the advantage over the unbiased AL baselines. In general, the performance on $B \in \{1, 5, 20\}$ show similar trends compared with the performance with batch size setting $B = 1$, which ensures the stability of our proposed at different batch size settings.

For different evaluation metrics, i.e., compare AUBC (acc) with AUBC (AUC) and AUBC (F_1), despite the difference in concrete AUBC values, the shape of the curves, the trend and the timing of model convergence are similar. That is, different evaluations also provide consistent results.

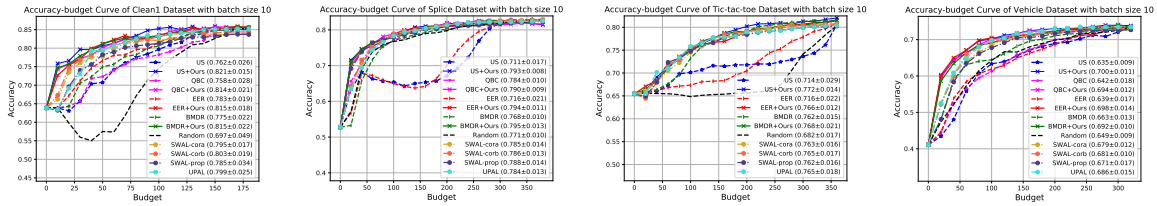
References

Alina Beygelzimer, Sanjoy Dasgupta, and John Langford. Importance weighted active learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 49–56, 2009.

Jorge Fernandez de Cossio and Jorge Fernandez de Cossio Diaz. Maximum entropy method: sampling bias, 2015.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31



(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

Figure 1: Accuracy-budget curves for classical ML tasks with $B = 10$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (acc) over 10 trials is shown in parentheses in the legend.

Sebastian Farquhar, Yarin Gal, and Tom Rainforth. On statistical bias in active learning: How and when to fix it. *arXiv preprint arXiv:2101.11665*, 2021.

Ravi Ganti and Alexander Gray. Upal: Unbiased pool based active learning. In *Artificial Intelligence and Statistics*, pages 422–431, 2012.

Henrik Imberg, Johan Jonasson, and Marina Axelson-Fisk. Optimal sampling in unbiased active learning. In *International Conference on Artificial Intelligence and Statistics*, pages 559–569. PMLR, 2020.

David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994.

N Roy and A McCallum. Toward optimal active learning through sampling estimation of error reduction. int. conf. on machine learning, 2001.

Christoph Sawade, Niels Landwehr, Steffen Bickel, and Tobias Scheffer. Active risk estimation. In *ICML*. Citeseer, 2010.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: acore-set approach. *stat*, 1050:27, 2017.

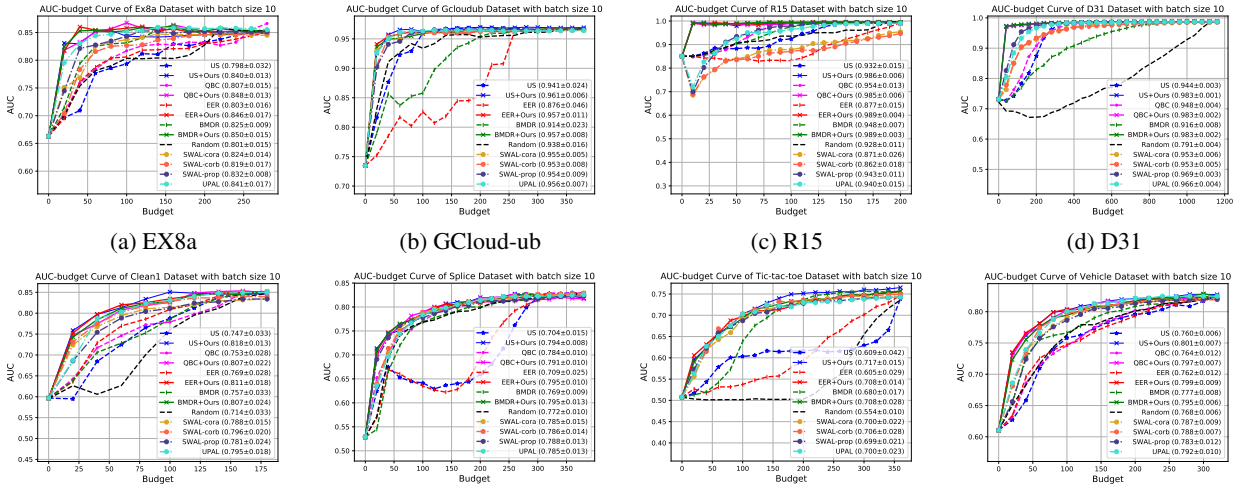
Burr Settles. Active learning literature survey. 2009.

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

Ying-Peng Tang, Guo-Xiang Li, and Sheng-Jun Huang. ALiPy: Active learning in python. Technical report, Nanjing University of Aeronautics and Astronautics, 2019. URL `https://github.com/NUAA-AL/ALiPy`. available as arXiv preprint `https://arxiv.org/abs/1901.03802`.

Zheng Wang and Jieping Ye. Querying discriminative and representative samples for batch mode active learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(3):1–23, 2015.

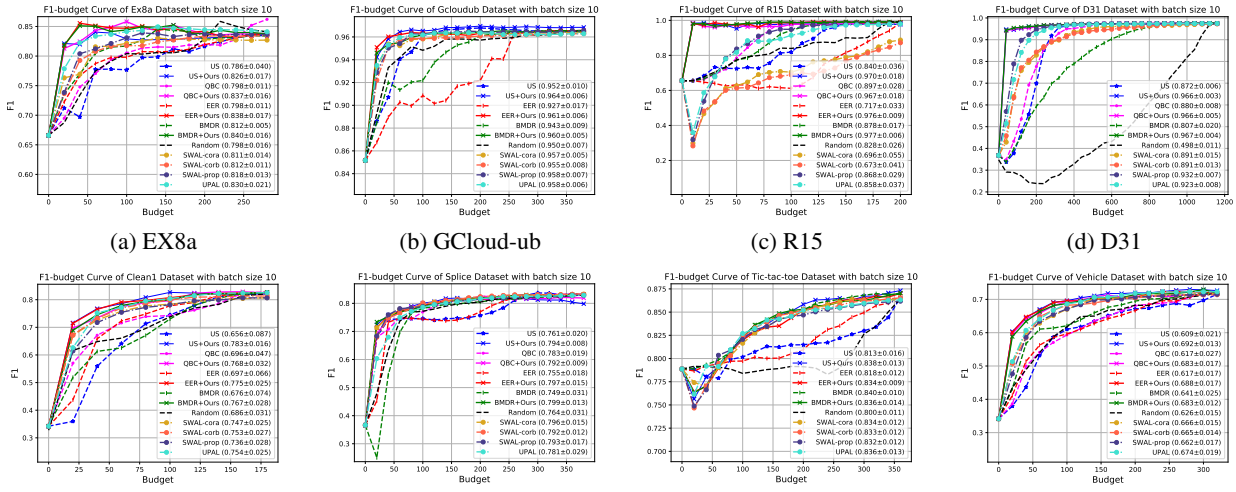
Yi Wu, Igor Kozintsev, Jean-Yves Bouguet, and Carole Dulong. Sampling strategies for active learning in personal photo retrieval. In *2006 IEEE International Conference on Multimedia and Expo*, pages 529–532. IEEE, 2006.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

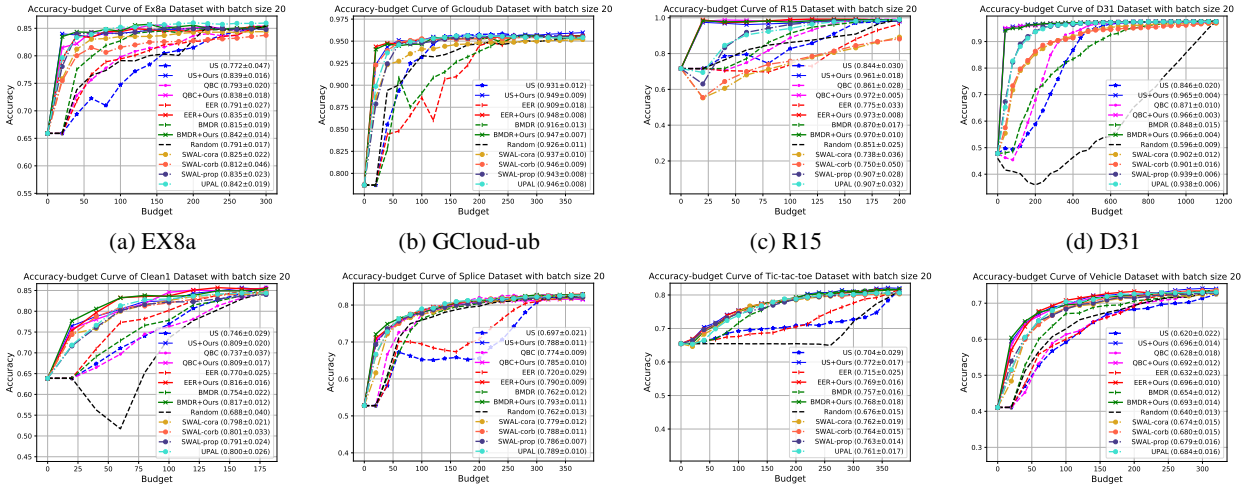
Figure 2: AUC-budget curves for classical ML tasks with $B = 10$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (auc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

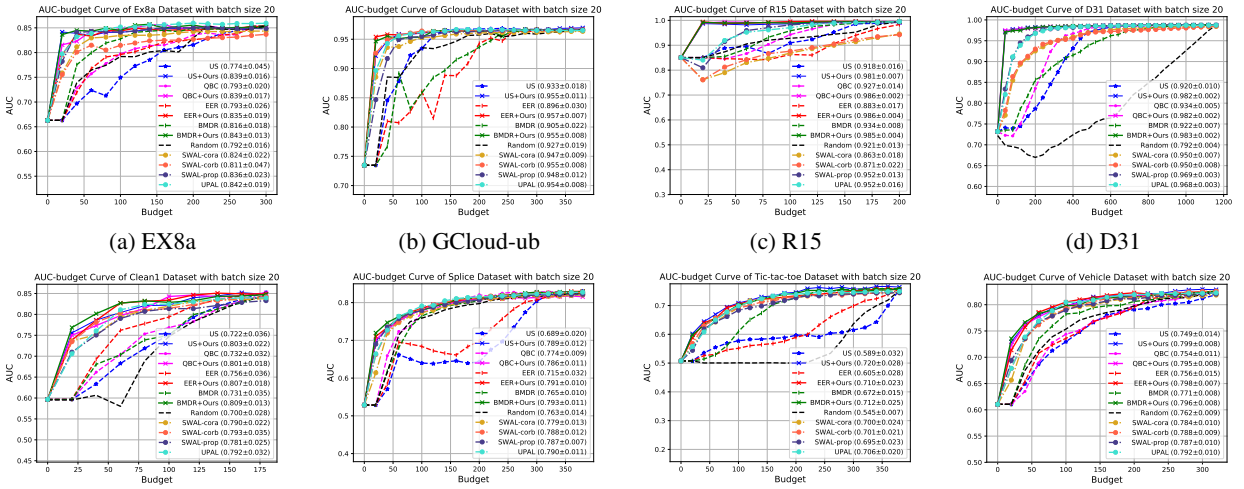
Figure 3: F_1-budget curves for classical ML tasks with $B = 10$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (f1) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

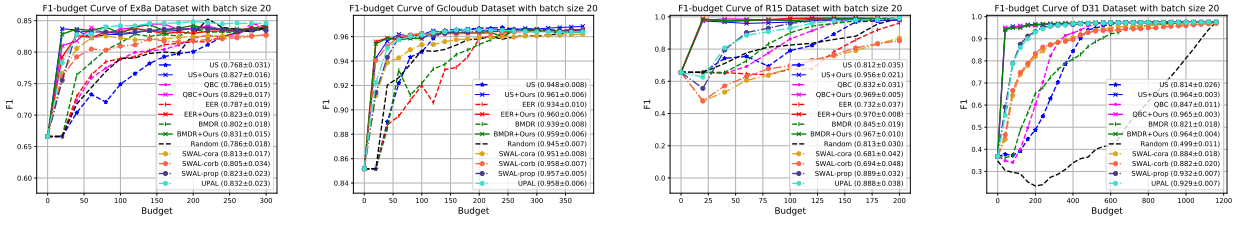
Figure 4: Accuracy-budget curves for classical ML tasks with $B = 20$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (acc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

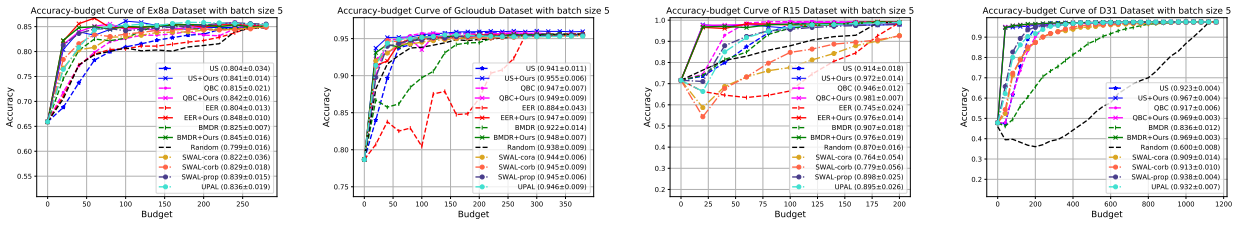
Figure 5: AUC-budget curves for classical ML tasks with $B = 20$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (auc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

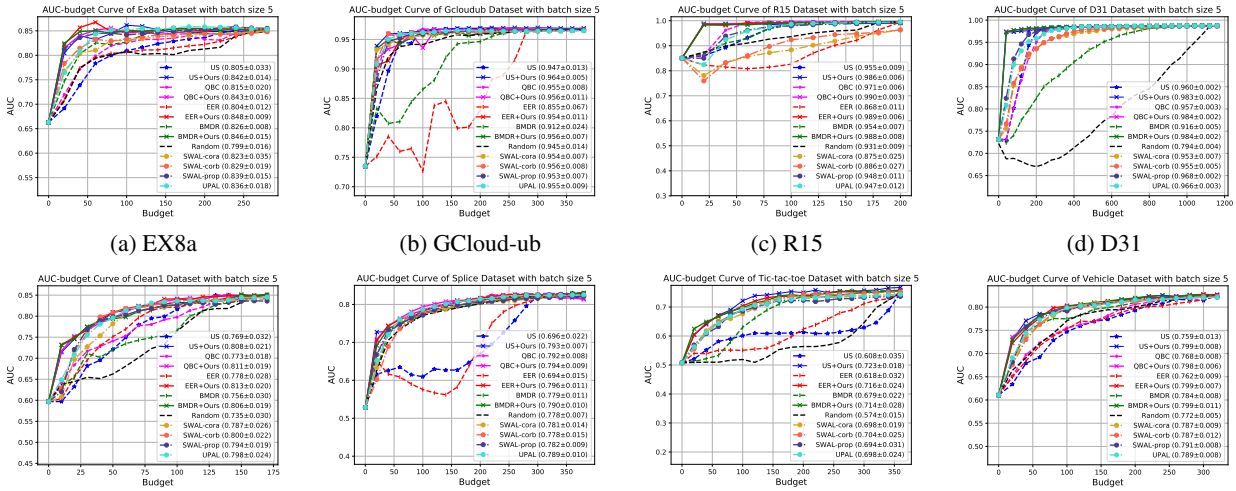
Figure 6: F_1-budget curves for classical ML tasks with $B = 20$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (f1) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

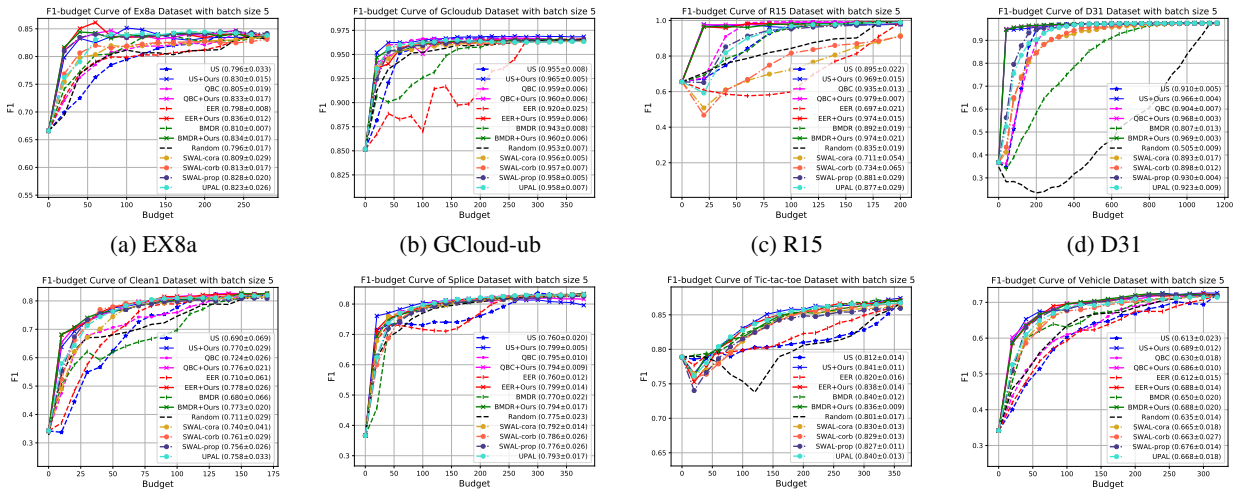
Figure 7: Accuracy-budget curves for classical ML tasks with $B = 5$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (acc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

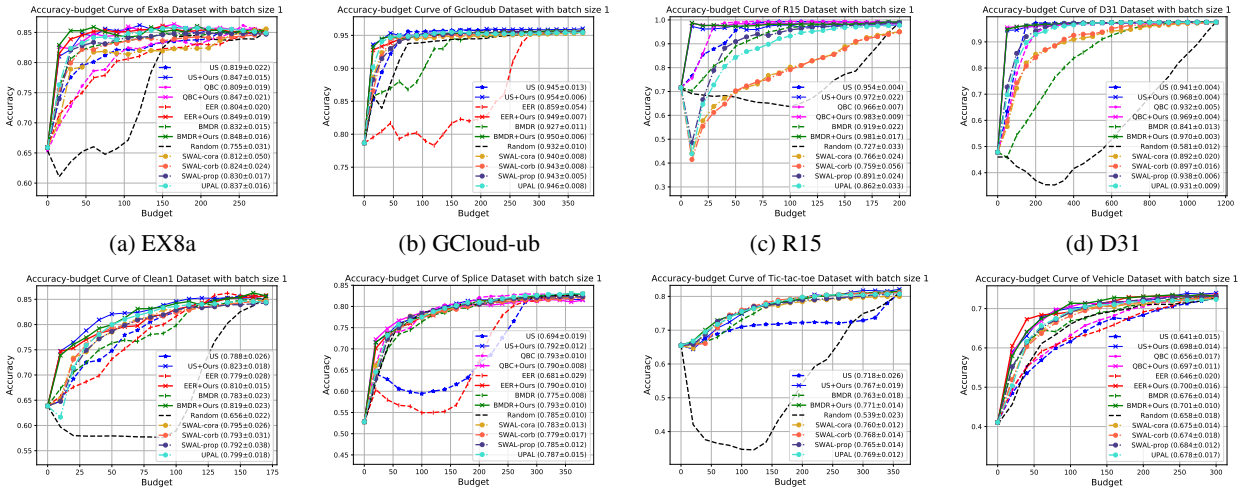
Figure 8: AUC-budget curves for classical ML tasks with $B = 5$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (auc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

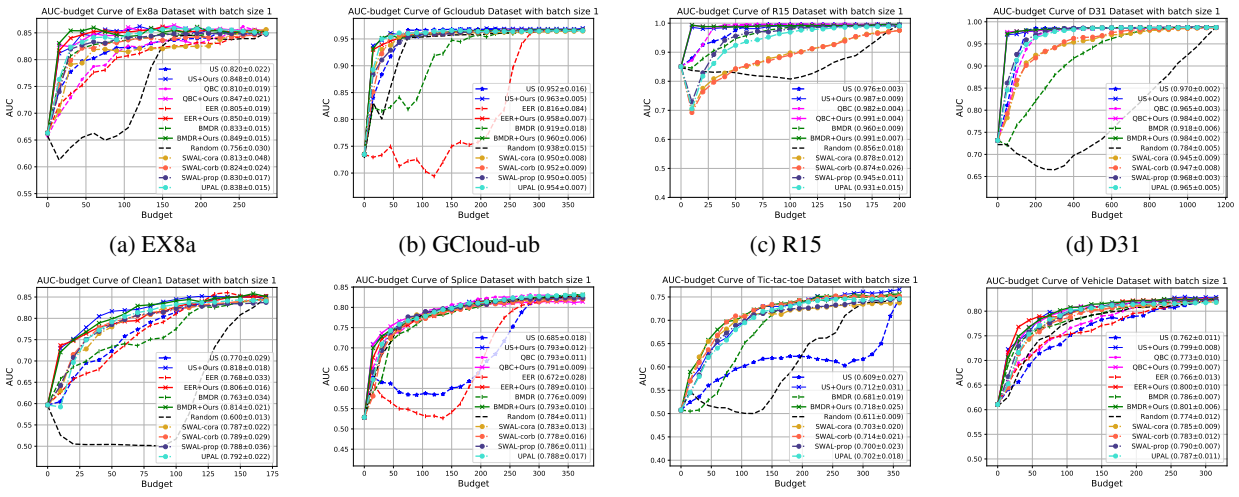
Figure 9: F_1-budget curves for classical ML tasks with $B = 5$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (f1) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

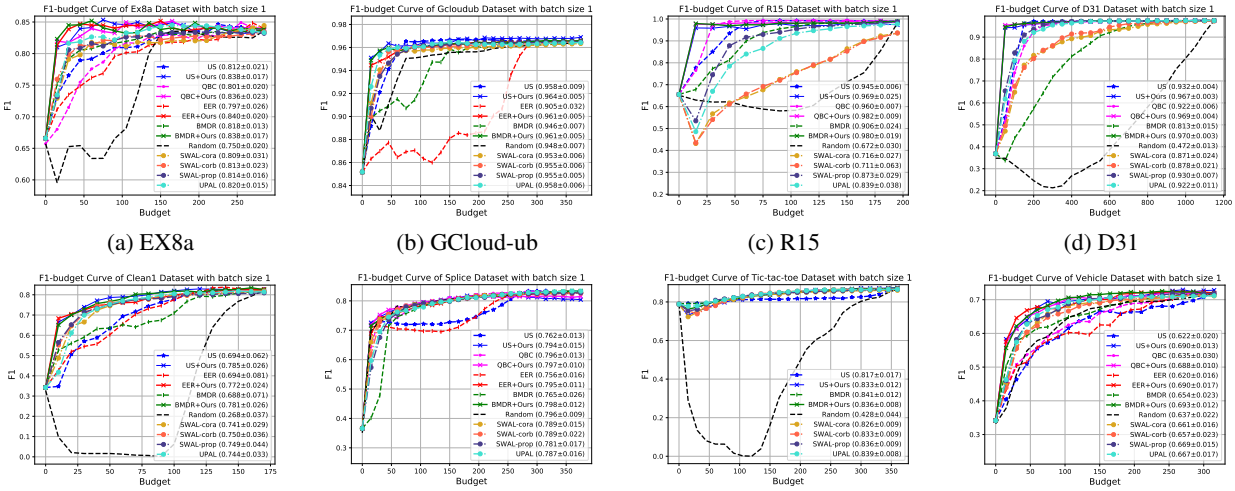
Figure 10: Accuracy-budget curves for classical ML tasks with $B = 1$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (acc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

Figure 11: AUC-budget curves for classical ML tasks with $B = 1$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (auc) over 10 trials is shown in parentheses in the legend.



(a) EX8a (b) GCloud-ub (c) R15 (d) D31

(e) Clean1 (f) Splice (g) Tic-tac-toe (h) Vehicle

Figure 12: F_1-budget curves for classical ML tasks with $B = 1$. The solid lines represent our proposed method and the dashed lines represent the corresponding baseline AL strategy. The mean and standard deviation of the AUBC (f1) over 10 trials is shown in parentheses in the legend.