Taming Small-sample Bias in Low-budget Active Learning

Linxin Song [1] **Jieyu Zhang** [2] **Xiaotian Lu** [3] **Tianyi Zhou** [4]

Abstract

Active learning (AL) aims to minimize the annotation cost by only querying a few informative examples for each model training stage. However, training a model on a few queried examples suffers from the small-sample bias. In this paper, we address this small-sample bias issue in low-budget AL by exploring a regularizer called Firth bias reduction, which can provably reduce the bias during the model training process but might hinder learning if its coefficient is not adaptive to the learning progress. Instead of tuning the coefficient for each query round, which is sensitive and time-consuming, we propose the curriculum Firth bias reduction (CHAIN) that can automatically adjust the coefficient to be adaptive to the training process. Under both deep learning and linear model settings, experiments on three benchmark datasets with several widely used query strategies and hyperparameter searching methods show that CHAIN can be used to build more efficient AL and can substantially improve the progress made by each active learning query.

1. Introduction

The last decade has witnessed the emergence of deep learning as the dominant force in advancing machine learning and its applications. To reduce the data annotation cost, researchers and practitioners have resorted to active learning (AL), which aims to reduce the total number of annotations by iteratively querying the most informative annotations for the learning process from an oracle (Settles, 2009; Schröder & Niekler, 2020). AL uses a finite budget to select and label a subset of a data pool for downstream supervised learning. By iteratively training a model on a current set of labeled data and querying labels for more new points, this paradigm yields models that use as fewer data as possible to achieve nearly the same accuracy as classifiers with unlimited labeling budgets.

Many traditional active learning approaches are based on uncertainty sampling (e.g., (Lewis, 1995; Scheffer et al., 2001; Culotta & McCallum, 2005; Joshi et al., 2009a; Li & Guo, 2013a; Gissin & Shalev-Shwartz, 2019; Sinha et al., 2019a)), wherein the learner queries the most uncertain examples for annotations, presumably because such examples are most informative to the learner. However, such a type of approach primarily relies on the quality of estimated uncertainty. Besides, existing deep active learning strategies require a large amount of labeled examples to work properly (Yuan et al., 2020; Pourahmadi et al., 2021). However, the labeling budget might be limited in practice. Under this low-budget regime, it is well-established that the conventional Maximum Likelihood Estimation (MLE) used for training the model is biased with a term of $O(N^{-1})$ (Cox & Snell, 1968; Box, 1971; Whitehead, 1986; Steyerberg et al., 1999). Such *small-sample bias* in low-budget setup might deteriorate the performance of uncertainty-based AL due to (1) an inaccurate model trained with MLE; and (2) a poor uncertainty estimation caused by the inaccurate model.

To tackle the small-sample bias in AL, we, for the first time, leverage the Firth bias reduction (Firth, 1993; Ghaffari et al., 2021) as a plug-in regularizer for the MLE-based model training. It reduces the small-sample bias by applying a regularizer in addition to the cross-entropy loss, which computes the log-determinant of Fisher information matrix $\log(\det(F))$. However, the performance of the Firth regularizer is sensitive to its coefficient (Ghaffari et al., 2021), a hyperparameter that controls the strength of the regularizer. Thus, how to search for the optimal coefficient becomes crucial when adopting the Firth regularizer. Grid search, as a standard solution for hyperparameter search, requires repeating the training process multiple times to search for the best coefficient. It is even more time-consuming in AL since the model has to be re-trained after each time the new queried data is added, and so does the hyperparameter search process.

To reduce the time complexity, we propose to optimize the Firth coefficient on the fly without heavy parameter tuning. Specifically, we formulate the optimization of both the

[1]Waseda University [2]University of Washington [3]Kyoto University [4]University of Maryland, College Park. Correspondence to: Linxin Song <songlx.imse.gt@ruri.waseda.jp>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

model parameters and the Firth coefficient as a bilevel optimization where the inner level optimization corresponds to the model parameters, and the outer one optimizes the Firth coefficient. Such a bilevel optimization can be solved efficiently by interleaving the gradient descent steps of the model parameter and the Firth coefficient. In experiments, it only requires a few gradient descent steps (*e.g.*, 50) to obtain a good coefficient.

Furthermore, the impact of the bias can vary drastically across different stages of the training process, while over-regularization might degrade the generalization performance, and under-regularization might not be able to alleviate the small-sample bias issue effectively. Hence, the strength of Firth regularization (*i.e.*, the Firth coefficient) should be adaptively adjusted rather than fixed. To this end, we propose a simple curriculum of the Firth coefficient that dynamically adjust the Firth coefficient to be adaptive to the model's training process.

Finally, we incorporate the aforementioned components seamlessly in CHAIN (**C**urriculum firt**H** bi**A**s Reduct**I**o**N**), which is a general query-strategy-agnostic method for tackling the small-sample bias issue in low-budget AL effectively and efficiently. We conduct extensive experiments on three benchmark datasets with five widely used query strategies under deep learning and linear model settings. Compared with the baselines, CHAIN achieves the highest performance in most of the query rounds. To help understand the curriculum of the Firth coefficient, we provide the analysis of the coefficient curriculum on both deep model and logistic regression, showing that the curriculum exists during training and is related to the size of the queried training set. We also conduct ablation studies for major hyperparameters used in CHAIN to demonstrate the consistent superiority of CHAIN under different hyperparameter choices.

Our main contributions are summarized as (1) we propose a query-strategy-agnostic framework that tackles the small-sample bias issue in low-budget AL via a plug-in regularization and efficient optimization of the coefficient of the regularizer; (2) we further improve the framework by a simple curriculum of the coefficient that adaptively changes the coefficient for the model in different training stages; (3) experimental results on three benchmark datasets with three widely used query strategies and two hyperparameter searching methods showing that CHAIN has a distinct and robust performance in most of the circumstances.

2. Related Works

Low-budget Active Learning. The goal of active learning (AL) is to render the best model given a limited budget of annotations. One of the fundamental components in AL is the query strategy, which selects a subset of instances from the pool of unlabeled data and queries their labels from an oracle (Settles, 2009). Existing query strategies can be categorized into uncertainty-based (Li & Guo, 2013b; Roth & Small, 2006; Wang & Shang, 2014), representation-based (Sener & Savarese, 2017b; Contardo et al., 2017; Yu et al., 2006) or hybrid (Sener & Savarese, 2017a; Shui et al., 2020; Sinha et al., 2019b). However, in the low-budget regime (e.g., less than 400 instances are labeled), the sub-optimality of the heuristic can lead to significant loss (Mahmood et al., 2021), which is called *cold-start* phenomenon. To tackle the *cold-start* issue, Mahmood et al. (2021) proposed a core set query strategy which minimized the discrete Wasserstein distance between the selected and unlabeled, while Hacohen et al. (2022) proposed a clustering algorithm for querying most typical instances from the max density region with diversity. Both of these methods partially alleviated the *cold-start* phenomenon by proposing new query strategies but overlooked the small-sample bias issue brought by using MLE in model training. In this work, instead of developing a new query strategy to compete with existing ones, we focus on the small-sample bias during model training and propose a general method that could improve over a given query strategy in low-budget AL.

Firth Bias Reduction. Firth bias reduction focuses on reducing the bias caused by maximum likelihood estimators while the dataset is small, known as Firths' Penalized Maximum Likelihood Estimator (PMLE) (Firth, 1993). In the case of the exponential family of distributions, it has a simplified form that penalizes the likelihood by Jeffrey's invariant prior (Firth, 1993; Poirier, 1994), which is proportional to the determinant of the Fisher Information Matrix F. For logistic and cosine classifiers (Chen et al., 2019), which are widely used in few-shot classification, since they belong to the exponential family, Firth bias reduction can be further cast as adding a log-determinant penalty ($\log(\det(F))$) to the cross-entropy loss. In this work, we, for the first time, explore the utility of Firth bias reduction in reducing the small-sample bias in low-budget AL.

3. Preliminary

Assume a K-way classification task and let $\mathcal{U} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ be the input unlabeled dataset with unknown ground truth $Y = [y_1, ..., y_N]$. Given a limited budget of annotations, the goal of active learning (AL) is to wisely spend the budget on querying annotations from an *oracle* (e.g., human expert) for model training. Specifically, we have $R > 1$ rounds of querying, and each time we query annotations for M instances according to some query strategy.

One of the popular query strategies is the uncertainty-based query, wherein different measurements of the model's uncertainty (e.g., entropy, the least confidence) on each instance can be leveraged. Given a specific query strategy Q, at round r we select the top-M uncertain instances from the remaining unlabeled instances. We include them in training set after obtaining their labels from the oracle (Li & Guo, 2013b; Roth & Small, 2006; Wang & Shang, 2014). The resultant training set $\mathcal{D}_r$ in round r is in turn used to train a classifier $f_r(\cdot;\boldsymbol{\beta}_r)$ parameterized by $\boldsymbol{\beta}_r$. We would omit the subscript r when there is no confusion. In this work, we focus on low-budget AL, where the budget is very low compared to the size of the pool of unlabeled data. In such a regime, each time model is trained with only a small set of labeled data, and usually with the Maximum Likelihood Estimation (MLE) (*i.e.*, the cross-entropy loss). As a consequence, the optimized model parameter $\boldsymbol{\beta}$ would be biased (Firth, 1993) and hurt both the model performance and the quality of the queries in the latter rounds, especially those uncertainty-based query strategies that largely rely on the trained model of last round to estimate the uncertainty of remaining unlabeled instances.

4. Methods

In this section, we describe the proposed method CHAIN (**C**urriculum firt**H** bi**A**s reduct**I**o**N**) that aims to tackle the small-sample bias issue in low-budget active learning. First, we show how to leverage the Firth bias reduction technique as a plug-in regularizer to reduce the small-sample bias in the model training procedure at each round of active learning. Then, we propose an efficient approach to optimize the coefficient of the Firth regularizer without heavy parameter tuning. Finally, we present a simple curriculum of the coefficient that adaptively adjusts the strength of the regularizer for the model at different training stages.

4.1. Firth Bias Reduction

Based on (Firth, 1993; Ghaffari et al., 2021), in this section, we introduce a penalized cross-entropy loss by adding the Firth regularizer to reduce the small-sample bias. We denote the parameter of the logistic model as $\boldsymbol{\beta}$, and the assignment probability of the i^{th} data to class k can be represented as

$$\mathbf{P}_{i,k} = \Pr(y_i = k|\mathbf{x}_i;\boldsymbol{\beta}) = \frac{\exp(\beta_k^\top \mathbf{x}_i)}{\sum_{k'=0}^{K}\exp(\beta_{k'}^\top \mathbf{x}_i)}. \tag{1}$$

The likelihood of the current training set $\mathcal{D}$ given the $\boldsymbol{\beta}$ is

$$\Pr(\mathbf{y}|\mathcal{D};\boldsymbol{\beta}) = \prod_{i=1}^{N}\sum_{k=0}^{K}\mathbb{1}[y_i = k]\cdot\mathbf{P}_{i,k}, \tag{2}$$

where $\mathbb{1}[\cdot]$ denotes the binary indicator function. We use the maximum likelihood estimation (MLE) to acquire the optimal parameter $\hat{\boldsymbol{\beta}}_{\text{MLE}}$, which can be defined as

$$\hat{\boldsymbol{\beta}}_{\text{MLE}} = \arg\max_{\boldsymbol{\beta}} \Pr(\mathbf{y}|\mathbf{x};\boldsymbol{\beta}). \tag{3}$$

Oftentimes the cross-entropy (CE) loss is used as the objective of the MLE, which can be interpreted as the log of Eq. 2. This draws out the optimization objective

$$\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{N} l_{\text{CE}}(y_i\|\mathbf{P}_{i,k}) = -\sum_{i=1}^{N}\sum_{k=0}^{K}\mathbb{1}[y_i = k]\log(\mathbf{P}_{i,k}). \tag{4}$$

In logistic models, the penalized likelihood function proposed by Firth is equivalent to imposing Jeffery's prior (Poirier, 1994) on the parameters and making a maximum posterior estimation as

$$\Pr(\boldsymbol{\beta}|\mathcal{D},\mathbf{y}) = \frac{1}{Z}\cdot\Pr(\mathbf{y}|\mathcal{D};\boldsymbol{\beta})\cdot\det(F|r)^{\frac{1}{2}} \tag{5}$$

where Z is a normalization constant and $\det(F|r)^{\frac{1}{2}}$ is the Jeffery's prior, the product of all r non-zero eigenvalues of the Fisher information matrix F to the power $1/2$, and F is defined as the Hessian of the negative log-likelihood function with

$$F = -\text{Hess}_{\boldsymbol{\beta}}(\mathcal{L}_{\text{CE}}) = \mathbb{E}_y[\nabla_{\boldsymbol{\beta}}\mathcal{L}_{\text{CE}}\cdot\nabla_{\boldsymbol{\beta}}\mathcal{L}_{\text{CE}}^\top]. \tag{6}$$

Taking the log of both sides of Eq.5 yields the log-posterior as a sum of the cross-entropy loss and the Firth bias reduction term

$$\mathcal{L}(\mathcal{D};\boldsymbol{\beta}) = \log(\Pr(\boldsymbol{\beta}|\mathcal{D},\mathbf{y})) = \mathcal{L}_{\text{CE}}(\mathcal{D};\boldsymbol{\beta}) + \lambda\cdot\mathcal{L}_{\text{Firth}}(\mathcal{D};\boldsymbol{\beta}), \tag{7}$$

where

$$\mathcal{L}_{\text{Firth}} = \frac{1}{2}\log(\det(F|r)) + \text{const}, \tag{8}$$

and λ is the coefficient of the regularizer. Following (Ghaffari et al., 2021), we further acquire an easy-to-implement optimization objective for the Firth regularizer. We firth decompose F as

$$F_{(dK)\times(dK)} = X^\top_{(dK)\times(NK)}\cdot M_{(NK)\times(NK)}\cdot X_{(NK)\times(dK)},$$

where d is the dimension of data feature space, M is a block-diagonal matrix whose i^{th} diagonal block is denoted as M_i. By adopting the Determinant Amendment and Constant Dropping (Ghaffari et al., 2021), and using the Matrix-Determinant Lemma (Harville, 1998), it comes out that

$$\begin{aligned}\mathcal{L}_{\text{Firth}} &= \frac{1}{N}\sum_{i=1}^{N}\log(\det(M_i)) + \text{Const} \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{k=0}^{K}\log(\mathbf{P}_{i,k}).\end{aligned} \tag{9}$$

It is well known that Jeffery's prior encourages uniform class assignment probabilities (Firth, 1993). Therefore, we treat the $\sum_{k=0}^{K} \log(\mathbf{P}_{i,k})$ as a scaled average over a uniform distribution of the number of classes

$$\mathcal{L}_{\text{Firth}} \propto \frac{1}{N} \sum_{i=1}^{N} \left[\sum_{k=0}^{K} \frac{1}{K+1} \cdot \log(\mathbf{P}_{i,k}) \right], \tag{10}$$

and it further comes out that

$$\mathcal{L}_{\text{Firth}} = \frac{-1}{N} \sum_{i=1}^{N} D_{\text{KL}}(\mathrm{U}_{[0,K]} \| \mathbf{P}_{i,k}) + \text{Const}, \tag{11}$$

where $\mathrm{U}_{[0,K]}$ is the uniform distribution over the number of classes. By combining with cross-entropy and dropping the constants, we have the final penalized optimization objective:

$$\mathcal{L} = \frac{-1}{N} \sum_{i=1}^{N} \left[l_{\text{CE}}(y_i \| \mathbf{P}_{i,k}) + \lambda \cdot D_{\text{KL}}(\mathrm{U}_{[0,K]} \| \mathbf{P}_{i,k}) \right]. \tag{12}$$

4.2. Bilevel Firth Coefficient Optimization

According to Eq. 12, we use λ as a coefficient to adjust the strength of the Firth regularizer. However, we found that the model performance is sensitive to the λ, because the variance of model performance with different λ is around 4 points on average. It indicates that one needs to perform some search algorithm to find the best λ. Furthermore, after each round of query, as new labeled instances are added and the size of the training set increases, we have to redo the search of the optimal λ, making the whole AL pipeline immensely time-consuming.

To attack this issue, we optimize λ without heavy parameter tuning via bilevel optimization. Specifically, let $\mathcal{D}$ and $\mathcal{V}$ be the current training set and a hold-out validation set, respectively. The bilevel optimization objective can be formulated as

$$\begin{aligned} \lambda^* &= \arg\min_{\lambda} \mathcal{L}_{\text{CE}}(\mathcal{V}; \boldsymbol{\beta}^*(\lambda)) \\ s.t.\ \boldsymbol{\beta}^*(\lambda) &= \arg\min_{\boldsymbol{\beta}} \mathcal{L}_{\text{CE}}(\mathcal{D}; \boldsymbol{\beta}) + \lambda \cdot \mathcal{L}_{\text{Firth}}(\mathcal{D}; \boldsymbol{\beta}). \end{aligned} \tag{13}$$

The above bilevel optimization problem is nontrivial to solve since computing the exact gradient of λ regarding the objective of the outer-level optimization requires an optimal solution of inner optimization. Specifically, Eq. 13 can be separated as the optimization of λ^* for which the best response $\boldsymbol{\beta}^*(\lambda)$ is required, and the optimization of the best response $\boldsymbol{\beta}^*(\lambda)$. For the optimization of $\boldsymbol{\beta}^*(\lambda)$, we adopt the SGD as an optimizer to update iteratively with time-step t

$$\boldsymbol{\beta}^{(t+1)} := \boldsymbol{\beta}^{(t)} - \eta \cdot \nabla_{\boldsymbol{\beta}^{(t)}} \mathcal{L}_{\text{T}}(\boldsymbol{\beta}^{(t)}, \lambda), \tag{14}$$

where η is the learning rate and $\mathcal{L}_{\text{T}}$ denotes the training loss. Assume we have T_2 steps totally for update the $\boldsymbol{\beta}$, the best response$\boldsymbol{\beta}^*(\lambda)$ can be represented as

$$\begin{aligned} \boldsymbol{\beta}^*(\lambda) &\approx \boldsymbol{\beta}^{(T_2)} - \eta \cdot \nabla_{\boldsymbol{\beta}^{(T_2)}} \mathcal{L}_{\text{T}}(\boldsymbol{\beta}^{(T_2)}, \lambda) \\ \text{where} \quad \boldsymbol{\beta}^{(T_2)} &:= \boldsymbol{\beta}^{(T_2-1)} - \eta \cdot \nabla_{\boldsymbol{\beta}^{(T_2-1)}} \mathcal{L}_{\text{T}}(\boldsymbol{\beta}^{(T_2-1)}, \lambda) \\ &\vdots \\ \text{where} \quad \boldsymbol{\beta}^{(1)} &:= \boldsymbol{\beta}^{(0)} - \eta \cdot \nabla_{\boldsymbol{\beta}^{(0)}} \mathcal{L}_{\text{T}}(\boldsymbol{\beta}^{(0)}, \lambda) \end{aligned} \tag{15}$$

To solve the optimization objective of λ, which requires the best response $\boldsymbol{\beta}^*(\lambda)$, we propose using an outer loop that wraps the optimization process for $\boldsymbol{\beta}^*(\lambda)$ described above. We seek a value λ^* which ensures that the training process $\mathcal{L}_{\text{T}}(\boldsymbol{\beta}, \lambda^*)$ produces parameter $\boldsymbol{\beta}^*(\lambda)$ which perform best against some metric $\mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda))$ we care about. Essentially, an estimate of $\boldsymbol{\beta}^*(\lambda)$ is obtained following T_2 steps training as outlined in Eq. 15, which is then evaluated against $\mathcal{L}_{\text{V}}$. The gradient of this evaluation, $\nabla_\lambda \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda))$ is then obtained through backpropagation, and used to update λ. Using $\tau = 1, ..., T_1$ as a time-step counter to symbolize the time-scale being different from that used in the *inner-loop*, we formalize this update as

$$\begin{aligned} \lambda^* &\approx \lambda^{(T_1)} - \gamma \cdot \nabla_{\lambda^{(T_1)}} \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda^{(T_1)}) \\ \text{where} \quad \lambda^{(T_1)} &:= \lambda^{(T_1-1)} - \gamma \cdot \nabla_{\lambda^{(T_1-1)}} \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda^{(T_1-1)})) \\ &\vdots \\ \text{where} \quad \lambda^{(1)} &:= \lambda^{(0)} - \gamma \cdot \nabla_{\lambda^{(0)}} \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda^{(0)})) \end{aligned} \tag{16}$$

where γ is the learning rate, $\mathcal{L}_{\text{V}}$ denotes the validation loss, and $\boldsymbol{\beta}^*(\lambda)$ can be solved by Eq. 15. Implementing an iterative process as described in Eq. 16 will exploit the chain rule for partial derivatives in order to run back propagation. By decompose the $\nabla_\lambda \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda))$, we have

$$\nabla_\lambda \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda)) = \nabla_{\boldsymbol{\beta}^{(t)}} \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda)) \cdot \nabla_\lambda \boldsymbol{\beta}^*(\lambda) \tag{17}$$

Obviously, $\nabla_{\boldsymbol{\beta}^{(t)}} \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda))$ exists for $t \in \{0, ..., T_2\}$. For $\nabla_\lambda \boldsymbol{\beta}^*(\lambda)$, the gradients trivially exist because: (1) λ is a continuous hyperparameter and (2) Eq. 16 is a differentiable function of λ. In this work, we use the GIMLI algorithm (Grefenstette et al., 2019) to implement the efficient calculation of $\nabla_\lambda \mathcal{L}_{\text{V}}(\boldsymbol{\beta}^*(\lambda))$.

4.3. The Curriculum of Firth Coefficient

Regarding the impact of the Firth coefficient on the model performance, it is natural to hypothesize that the model

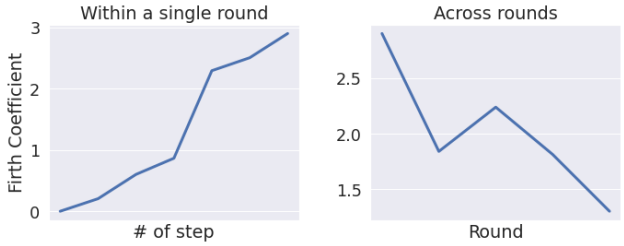


Figure 1: Example of the Firth coefficient curriculum. *Left* is the coefficient curve in a single model training. *Right* is the curve of the final coefficient of a model training across AL query rounds.

at different training stages may require different λ to obtain the best performance since the model may accumulate different levels of bias during the training. Therefore, we propose a simple curriculum of the Firth coefficient. That is, we repeatedly update the λ via, again, solving the bilevel optimization but regarding the currently trained model. Specifically, for every T_2 model training step, we use the current model parameters as the initialization of the $\boldsymbol{\beta}$ in the inner optimization of Eq. 13 and then solve the bilevel optimization to get the new λ. In this way, the λ is specific to the currently trained model and dynamically changes over the training process.

As in Fig. 1, the coefficient we obtained by solving the bilevel optimization is increasing within a single model training. However, the final coefficient (the very last coefficient of a model training process) across different AL query rounds decreases with the increase in the size of the training set. This phenomenon provides some insights into using a curriculum of λ: (1) within one model training, the bias is gradually increasing as the model saw a small set of training data multiple times, and therefore the λ increases as well to strengthen the regularizer; (2) when we have more and more training data as the AL proceeds, the small-sample bias issue is naturally alleviated so the final λ decreases. Finally, the overall algorithm of CHAIN is summarized in Alg. 1.

Algorithm 1 CHAIN

Input: Unlabeled dataset $\mathcal{U}$; Query strategy Q; Size of each query M.

$\mathcal{D}_1 \leftarrow$ Randomly sample M instances from $\mathcal{U}$ and query the labels.
$\mathcal{U}_1 \leftarrow \mathcal{U} \setminus \mathcal{D}_1$
for $r = 1, 2, ..., R$ **do**
  for $t = 1, 2, ..., T$ **do**
    if $t \; mod \; T_2 = 0$ **then**
      $\lambda_t^* \leftarrow$ solve Eq. 13.
    $\boldsymbol{\beta}^{(t+1)} \leftarrow$ perform a gradient descent step.
  $\boldsymbol{\beta}_r^* \leftarrow \boldsymbol{\beta}^{(T)}$.
  $\mathcal{S}_{r+1} \leftarrow Q(\text{from} : \mathcal{U}_r, \text{size} : M, \text{model} : f(\cdot; \boldsymbol{\beta}_r^*))$.
  Query annotations of $\mathcal{S}_{r+1}$ from Oracle.
  $\mathcal{D}_{r+1} \leftarrow \mathcal{D}_r \cup \mathcal{S}_{r+1}$.
  $\mathcal{U}_{r+1} \leftarrow \mathcal{U}_r \setminus \mathcal{D}_{r+1}$.

Output: Output final model $f(\cdot; \boldsymbol{\beta}_R^*)$.

5. Experiment

In this section, we summarize the details of the experiment setting, including the datasets, AL setups, implementation details, and baselines. We further provide the comparison result, ablation study, and Firth coefficient curriculum study on fine-tuning and logistic regression tasks.

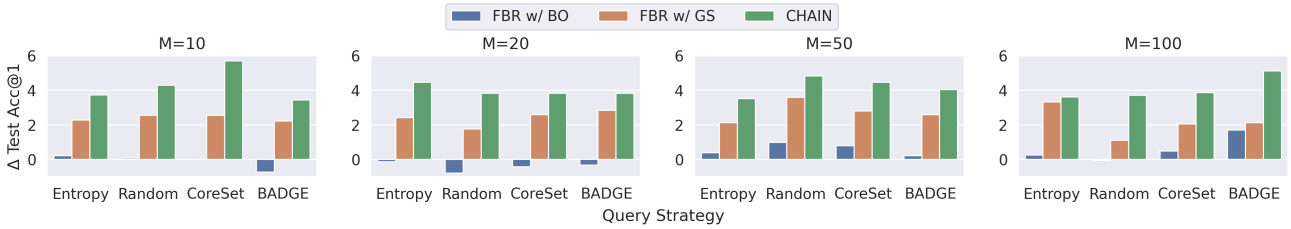
5.1. Experiment Setups

Datasets and AL setups. We use three image classification benchmark datasets (CIFAR10 (Krizhevsky et al., 2009), CIFAR100 (Krizhevsky et al., 2009) and Fashion MNIST (Xiao et al., 2017)) to evaluate CHAIN. The datasets are separated into training and test sets originally, and we randomly sample a small-size validation set from the training set. We study both deep learning and linear model settings. For the former, we fix the model being trained to be ResNet-18 (He et al., 2016) pre-trained on ImageNet (Deng et al., 2009), while for the latter, we use the features extracted from also a ResNet-18 pre-trained on ImageNet and fit a logistic regression model on top of the extracted features. To demonstrate the efficacy of CHAIN for reducing the small-sample bias in low-budget AL, we set our total budget as $500 = M \times R$, where the budget of the training set at each query round is $M \in \{10, 20, 50, 100\}$, and total query round $R = 500/M$.

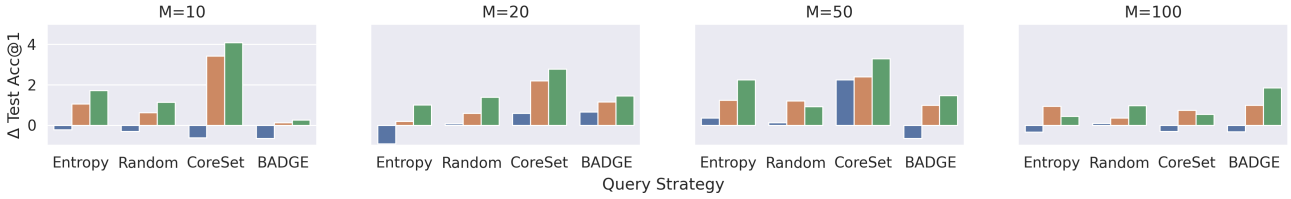
Implementations. In the proposed CHAIN, we have two major hyperparameters: T_1 is the number of steps we used in solving the bilevel optimization, and within one model training, we solve the bilevel optimization to get a new λ every T_2 model training steps. For the deep learning setting, we set $T_1 = 50$ and $T_2 = 100$ for all datasets. For logistic regression, we set $T_1 = 1$ and $T_2 = 5$ for all the datasets. We use SGD with a momentum of 0.9 for fine-tuning the ResNet-18 and Adam for training the logistic regression model. We use 0.01 as the learning rate for fine-tuning and 0.001 for logistic regression. For bilevel optimization, we use Adam as our optimizer, and we set the learning rate as 0.01 for the deep learning setting and 0.05 for the linear model setting. To reduce the impact of changing the training set to the AL optimization process, we keep our batch size $|\mathcal{B}|$ positive proportional to the training set size $|\mathcal{S}_t|$, *i.e.*, $|\mathcal{B}| = 0.1|\mathcal{S}_t|$. All our experiments are conducted on 6 Nvidia A6000 GPUs and Intel Xeon Gold

Table 1: Final round performance comparison with and without CHAIN of fine-tunning ResNet18 on CIFAR10, CIFAR100 and Fashion MNIST. Blue denotes the best result in each M, and red denotes the best results over all query strategies and M in the corresponding dataset. Performance overall rounds are listed in the appendix.

Dataset(↓)	Methods(↓)	M=10				M=20			
		Entropy	CoreSet	BADGE	Random	Entropy	CoreSet	BADGE	Random
CIFAR10	Orig.	52.26	52.84	53.04	56.06	51.88	52.61	53.26	54.43
	w/ CHAIN	57.65	59.56	58.03	60.15	55.90	56.90	56.94	57.47
CIFAR100	Orig.	15.52	14.63	16.50	15.63	15.55	15.01	15.77	15.79
	w/ CHAIN	17.13	16.12	17.82	18.08	16.62	16.36	16.21	18.82
Fashion MNIST	Orig.	88.50	89.44	87.47	90.08	90.72	89.93	88.64	90.04
	w/ CHAIN	91.08	93.41	92.22	91.28	90.99	91.36	91.64	91.48
Dataset(↓)	Methods(↓)	M=50				M=100			
		Entropy	CoreSet	BADGE	Random	Entropy	CoreSet	BADGE	Random
CIFAR10	Orig.	53.57	53.51	54.23	52.25	52.26	53.58	52.12	54.01
	w/ CHAIN	58.49	57.14	58.39	58.63	57.48	57.37	58.56	58.74
CIFAR100	Orig.	14.52	15.28	16.38	16.63	14.49	15.68	16.76	15.82
	w/ CHAIN	15.95	17.25	17.58	16.83	14.69	16.05	17.51	17.70
Fashion MNIST	Orig.	89.20	88.32	91.88	90.43	90.32	90.13	90.11	89.68
	w/ CHAIN	92.12	90.94	92.78	90.90	90.09	90.65	93.47	91.16



(a) CIFAR10



(b) CIFAR100

Figure 2: Comparison of the performance gain over all rounds between FBR+BO, FBR+GS and CHAIN of fine-tunning ResNet18 on CIFAR10, CIFAR100. Both of the sub-figures share the same figure legend shown in Fig.2a. Results on Fashion MNIST are listed in the appendix.

6226R CPU, and our code[1] is implemented in Python 3.8 and PyTorch 1.9.

5.2. Baselines

CHAIN can be readily combined with any query strategy since it only modifies the model training procedure. To demonstrate the above claim, we choose three well-known uncertainty-based, representation-based, and hybrid query strategies and the naive random query strategy to show that CHAIN can improve the model performance by reducing the small-sample bias in low-budget AL. Besides those selected query strategies, we also compare CHAIN with two widely used hyperparameter searching methods to show that CHAIN performs better without efficiency loss.

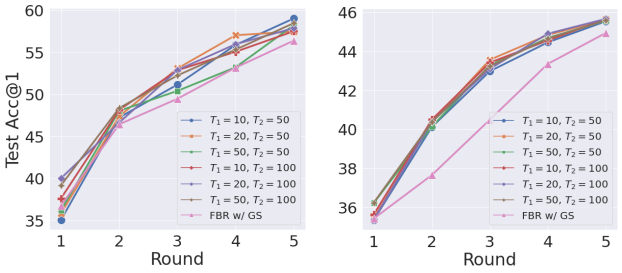
5.2.1. Query Strategies

Entropy (Joshi et al., 2009b) Entropy is the most typical uncertainty-based query strategy. Higher entropy values indicate greater uncertainty in the probability distribution. **CoreSet** (Sener & Savarese, 2017c) is a representation-based query strategy that adopts the coreset selection based on data embeddings(representations). **BADGE** (Ash et al., 2019) is a hybrid query strategy incorporating both predic-

[1]We will release the source code after the paper is accepted.

Table 2: Final round performance comparison with and without CHAIN of logistic regression on CIFAR10. blue denotes the best result in each M, and red denotes the best results over all query strategies and M in the corresponding dataset. Performance overall rounds are listed in the appendix.

Dataset(↓)	Methods(↓)	M=10				M=20			
		Entropy	CoreSet	BADGE	Random	Entropy	CoreSet	BADGE	Random
CIFAR10	Orig.	28.28	11.62	43.03	39.33	31.27	10.56	40.18	40.96
	w/ CHAIN	37.47	32.15	44.12	43.44	37.79	33.90	44.86	41.36
Dataset(↓)	Methods(↓)	M=50				M=100			
		Entropy	CoreSet	BADGE	Random	Entropy	CoreSet	BADGE	Random
CIFAR10	Orig.	33.50	11.49	44.06	40.35	28.35	11.51	42.04	39.87
	w/ CHAIN	36.79	34.29	44.93	42.19	39.61	37.21	45.61	42.99



(a) Fine-tuning a ResNet18 (b) Logistic Regression

Figure 3: Ablation studies of the two hyperparameters T_1 and T_2 on CIFAR10 dataset with random query strategy and $M = 100$. All the results are averaged over three runs.

tive uncertainty and data diversity into every selected batch.

5.2.2. Hyperparameter searching methods

To show the efficacy of our bilevel optimization approach for seeking the best Firth coefficient, we include the Firth bias reduction (FBR) with grid search over the coefficient as a baseline (marked as **FBR w/ GS**, GS: Grid Search). To further verify the usefulness of the proposed curriculum of the coefficient, our last baseline is **FBR w/ BO** (BO: Bilevel Optimization), which only applies the bilevel optimization once before the model training process and uses the output λ as a fixed hyperparameter during the model training.

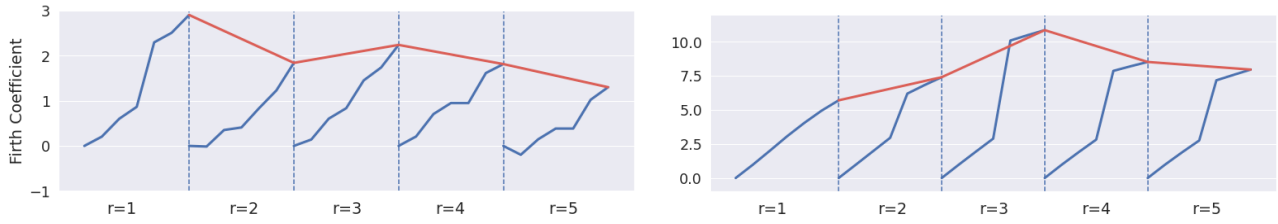
5.3. Main Result on Fine-tuning

We compared CHAIN with all baseline methods across four query strategies on three benchmark datasets. We record the average performance of every query round over three runs. The comparison at the final round is presented in Tab. 1. The results clearly demonstrate that adopting CHAIN improves the performance of the selected query strategies. To further investigate the significance of the observed performance improvement, we conducted a t-test with the null hypothesis $\mu_{\text{Orig.}} = \mu_{\text{w/ CHAIN}}$ and the alternative hypothesis $\mu_{\text{Orig.}} < \mu_{\text{w/ CHAIN}}$. The resulting t-values for the performance between the two selected methods over CIFAR10, CIFAR100, and Fashion MNIST are $-15.52, -5.24$, and -6.19, respectively. All of these values are smaller than the critical value of $-t(N_{\text{Orig.}} + N_{\text{w/ CHAIN}} - 2, 0.10) = -1.6859$, providing strong evidence to reject the null hypothesis in favor of the alternative hypothesis. Further analysis reveals that the best performance for CIFAR10 and CIFAR100 is achieved with a small M, while for Fashion MNIST, the best performance is observed with a large M (as shown in the results highlighted in red in Tab. 1). This can be attributed to the small-sample bias, which is more significant in CIFAR10 and CIFAR100 due to their complex features compared to Fashion MNIST, which has relatively simple features. Therefore, the value of λ at the beginning of the training process is generally larger for CIFAR10 and CIFAR100, while it is smaller for Fashion MNIST (as discussed in Sec. 5.6, and we put the further discussion in appendix).

We also compared the over-round performance gain of our method with other well-known hyperparameter searching methods across all datasets, and we recorded the results in Fig. 2. Upon analyzing the results of CHAIN, we observed that the performance per round was relatively consistent overall M for entropy and random query strategies. However, for CoreSet and BADGE, the performance gain displayed a decreasing trend (for CoreSet) and an increasing trend (for BADGE), respectively. This indicates that the CoreSet strategy is highly biased when the dataset is small, while the bias of BADGE increases as the dataset grows larger. Therefore, we recommend using BADGE with lower M and CoreSet with higher M.

5.4. Main Result on Logistic Regression

To theoretically demonstrate the effectiveness of CHAIN, we evaluated CHAIN and other baseline methods using a logistic regression model on CIFAR10. The results are presented in Tab. 2, from which we can observe that CHAIN



(a) Fine-tuning a ResNet-18

(b) Logistic Regression

Figure 4: The trend of the Firth coefficient across model training steps and AL query rounds on CIFAR10 with random query strategy and $M = 100$.

consistently improves the overall performance across all query strategies. Using the same t-test as in Sec. 5.3, we obtained a t-value of -3.32, which is smaller than the critical value of $-t(N_{\text{Orig.}} + N_{\text{w/ CHAIN}} - 2, 0.10) = -1.6859$. Based on this finding, we can conclude that the adoption of CHAIN leads to a significant improvement in performance. Upon further examining Tab. 1, we observed that CoreSet failed to converge to good performance with logistic regression. However, a significant improvement was observed after adopting CHAIN, indicating that the bias of CoreSet is amplified with less-parameterized models such as logistic regression. On the other hand, BADGE has a strong bias-restricting ability, enabling it to achieve better performance even with less-parameterized models. Further discussion of the performance of logistic regression can be found in the appendix.

5.5. Ablation Study

We provide ablation studies on two major hyperparameters: T_1 and T_2. Tuning T_1 will influence the bilevel optimization, which determines the quality of the learned coefficient, while tuning the T_2 affects the frequency of the coefficient being updated during the model training. We run CHAIN with varying $T_1 \in \{10, 20, 50\}$ and $T_2 \in \{50, 100\}$, and compare it with FBR w/ GS. The results can be found in Fig. 3. Following the results, CHAIN outperforms grid search in all combinations of T_1 and T_2. We can see that performance of CHAIN with different hyperparameters is more consistent with lower variance when using the logistic regression model than the deep model. This observation indicates that the complex deep model is relatively more sensitive to the bilevel optimization process and the curriculum of the Firth coefficient.

5.6. The Curriculum of the Firth Coefficient

To study the curriculum of the coefficient during training, we record the optimized coefficient during each single model training and across different AL rounds, as summarized in Fig. 4 (results of CIFAR100 and Fashion MNIST are recorded in the appendix). It is obvious that in both cases of fine-tuning a deep model and training a logistic regression model, the coefficient increases within each round, and the curve gradually becomes flat. The reason for the increasing coefficient is that at the very beginning of the training, the model is randomly initialized and thus contains no bias, but as the training proceeds, the model is gradually affected by the small-sample bias, so the strength of the regularization should increase as well. And we hypothesize that the reason for the flat coefficient curve at the final stage is that the change in the model parameters becomes relatively minor when the model is well-trained. Therefore the optimal coefficient remains similar for a similar model.

In terms of the dynamic of the coefficient across AL rounds, the final step coefficient shows a decreasing trend in fine-tuning a deep model, while in the case of the logistic regression model, the coefficient increases in the early rounds but decreases in the later rounds. For the deep learning setting, the final coefficient decreases as expected because as we include more labeled data, the small-sample bias issue is naturally alleviated. More interestingly, for the logistic regression model, the increasing trend of the final coefficient in the early rounds may seem counter-intuitive since, with more training data, we expect the strength of Firth regularization to decrease. We think the underlying reason is that in the early query rounds, we observed that the logistic regression model would trigger the early stopping in fewer training steps than that of later rounds, so the final coefficient is lower than in later rounds wherein the model is trained with more steps, and the coefficient is updated more times.

6. Conclusion

In this work, we focus on the small-sample bias issue in low-budget active learning. To mitigate the small-sample bias, we leverage the Firth bias reduction. Furthermore, we introduce a simple curriculum of the coefficient, which can adjust the regularization strength according to the current model training stage. The experimental results show that CHAIN outperforms all baselines in most cases.

References

Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J., and Agarwal, A. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.

Box, M. J. Bias in nonlinear estimation. *Journal of the royal statistical society series b-methodological*, 1971.

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., and Huang, J.-B. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019.

Contardo, G., Denoyer, L., and Artières, T. A meta-learning approach to one-step active learning. *arXiv preprint arXiv:1706.08334*, 2017.

Cox, D. and Snell, E. J. A general definition of residuals. *Journal of the royal statistical society series b-methodological*, 1968.

Culotta, A. and McCallum, A. Reducing labeling effort for structured prediction tasks. *national conference on artificial intelligence*, 2005.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Firth, D. Bias reduction of maximum likelihood estimates. *Biometrika*, 80(1):27–38, 1993.

Ghaffari, S., Saleh, E., Forsyth, D., and Wang, Y.-X. On the importance of firth bias reduction in few-shot classification. In *International Conference on Learning Representations*, 2021.

Gissin, D. and Shalev-Shwartz, S. Discriminative active learning. *arXiv: Learning*, 2019.

Grefenstette, E., Amos, B., Yarats, D., Htut, P. M., Molchanov, A., Meier, F., Kiela, D., Cho, K., and Chintala, S. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*, 2019.

Hacohen, G., Dekel, A., and Weinshall, D. Active learning on a budget: Opposite strategies suit high and low budgets. *arXiv preprint arXiv:2202.02794*, 2022.

Harville, D. A. Matrix algebra from a statistician's perspective, 1998.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Joshi, A., Porikli, F., and Papanikolopoulos, N. Multi-class active learning for image classification. *computer vision and pattern recognition*, 2009a.

Joshi, A. J., Porikli, F., and Papanikolopoulos, N. Multi-class active learning for image classification. In *2009 ieee conference on computer vision and pattern recognition*, pp. 2372–2379. IEEE, 2009b.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Lewis, D. D. A sequential algorithm for training text classifiers: Corrigendum and additional data. In *Acm Sigir Forum*, volume 29, pp. 13–19. ACM New York, NY, USA, 1995.

Li, X. and Guo, Y. Adaptive active learning for image classification. *computer vision and pattern recognition*, 2013a.

Li, X. and Guo, Y. Adaptive active learning for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 859–866, 2013b.

Lorraine, J., Vicol, P., and Duvenaud, D. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics*, pp. 1540–1552. PMLR, 2020.

Mahmood, R., Fidler, S., and Law, M. T. Low budget active learning via wasserstein distance: An integer programming approach. *arXiv preprint arXiv:2106.02968*, 2021.

Poirier, D. Jeffreys' prior for logit models. *Journal of Econometrics*, 63(2):327–339, 1994.

Pourahmadi, K., Nooralinejad, P., and Pirsiavash, H. A simple baseline for low-budget active learning. *arXiv: Computer Vision and Pattern Recognition*, 2021.

Roth, D. and Small, K. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pp. 413–424. Springer, 2006.

Scheffer, T., Decomain, C., and Wrobel, S. Active hidden markov models for information extraction. *Intelligent Data Analysis*, 2001.

Schröder, C. and Niekler, A. A survey of active learning for text classification using deep neural networks. *arXiv preprint arXiv:2008.07267*, 2020.

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

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

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

Settles, B. Active learning literature survey. 2009.

Shui, C., Zhou, F., Gagné, C., and Wang, B. Deep active learning: Unified and principled method for query and training. In *International Conference on Artificial Intelligence and Statistics*, pp. 1308–1318. PMLR, 2020.

Sinha, S., Ebrahimi, S., and Darrell, T. Variational adversarial active learning. *international conference on computer vision*, 2019a.

Sinha, S., Ebrahimi, S., and Darrell, T. Variational adversarial active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5972–5981, 2019b.

Steyerberg, E. W., Eijkemans, M. J., and Habbema, J. D. F. Stepwise selection in small data sets : A simulation study of bias in logistic regression analysis. *Journal of Clinical Epidemiology*, 1999.

Wang, D. and Shang, Y. A new active labeling method for deep learning. In *2014 International joint conference on neural networks (IJCNN)*, pp. 112–119. IEEE, 2014.

Whitehead, J. On the bias of maximum likelihood estimation following a sequential test. *Biometrika*, 1986.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

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

Yuan, M., Lin, H.-T., and Boyd-Graber, J. Cold-start active learning through self-supervised language modeling. *empirical methods in natural language processing*, 2020.

A. Extra Experiment

We provide the per-round comparison in Fig. 5, and the final round results are reported in the main context. To clarify the figure, we omit the curve of w/ GS and w/ BO of all query strategies and only report the final round results in Tab.3. We can see from the results that in most circumstances, training with a curriculum coefficient provides a promising performance gain over all rounds in CIFAR10 and CIFAR100, which is also reported in Fig. 2a.

A.1. Other hyperparameter searching methods

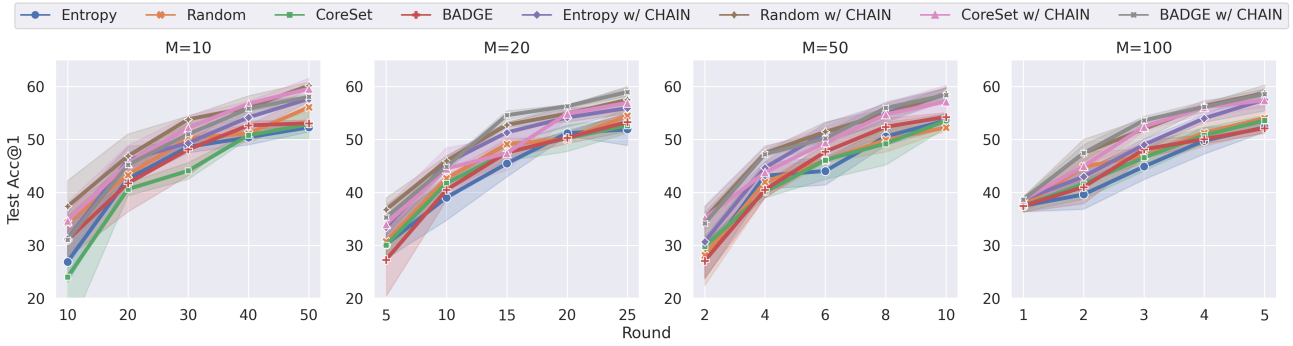
From Tab. 3, we can find that FBR+GS also performs better than the origin under most of the situations but less than CHAIN because of the limited searching space. In this work, we search the λ in $\{0.0, 0.01, 0.1, 1.0, 3.0\}$ in each round, which means we will train five different models from scratch in each round, comparing their performance, and choose the best one to query data by the baseline query strategies from the dataset. While FBR+GS yields good results, supporting such a search process requires huge performance, especially in active learning, which has multiple rounds, and we need to train a model from scratch at each round to prevent the influence of changing the distribution of the training set. On the other hand, FBR+BO performs less than the original in many circumstances. This is because bi-level optimization has a huge noise when the model is under-fitting, which is why we require T_2 steps in CHAIN to get a "best response" of λ (Lorraine et al., 2020).

A.2. Less perform in Fashion MNIST

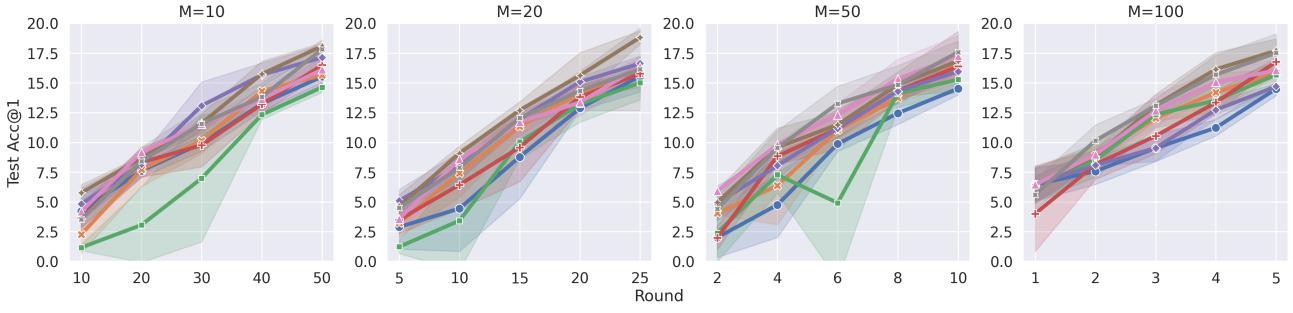
In the Fashion MNIST, the over-round performance gain of CHAIN is sometimes less than FBR w/ GS, but CHAIN still achieves the best final-round result in most circumstances. By deep into the curriculum of the λ of CHAIN when performing on Fashion MNIST (Fig. 6), we observe that CHAIN tends to learn a negative value of λ, which according to Eq. 12 means that the model tends to "sharpen" the output confidence by minimizing the KL-divergence between the uniform distribution and the output confidence. This observation indicates that the model can capture the features of Fashion MNIST under a low-budget regime with only a few steps, such that the model's performance on the training set can be improved by giving a more confident output (reflected in a sharp confidence distribution). However, such an optimization process will speed up the over-fitting, leading to less performance of CHAIN at the start of training. Fortunately, the "over-fitting" process can be mitigated by CHAIN in the later round, which can be observed in Fig. 7b later round. Moreover, this can be further alleviated by increasing the query rounds so that the CHAIN achieve the best average performance in $M = 10$ on Fashion MNIST.

A.3. CoreSet versus BADGE in logistic regression

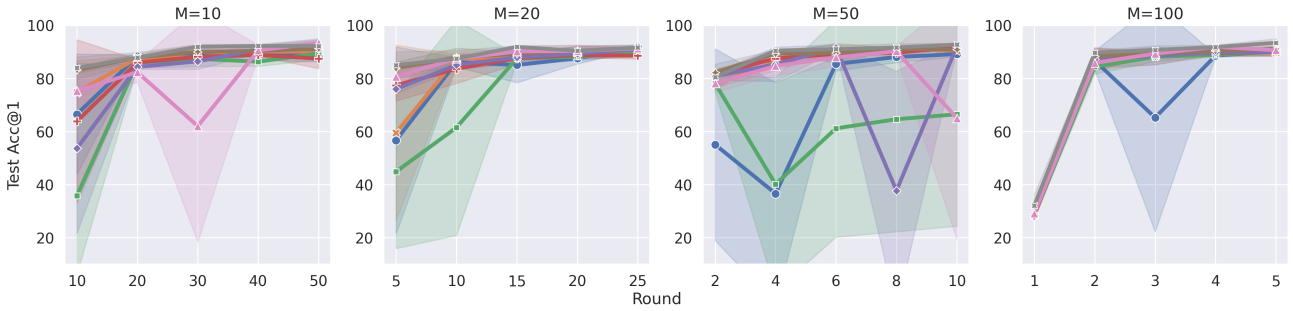
We found that in logistic regression, the CoreSet cannot converge in CIFAR10 overall situations, and the bad performance appears consistently. This is because CoreSet is a representation-based query strategy, and in the logistic regression, we froze the parameter of the feature extractor (ResNet18), which means the representation is fixed during training. As a result, CoreSet will always query data with noised embeddings, leading to a bad performance. While in the first round, the data is queried randomly to eliminate the bias of dataset initialization, so the coreset achieves better performance in the first round, and the performance gain is eliminated by the later selected noise data points. However, BADGE does not suffer from the invariant embeddings even though it leverages the same embeddings of CoreSet. This is because BADGE encodes the gradient, a representation of uncertainty, of the current model (logistic regression) into the embeddings, which helps the model query the data with large uncertainty, which can provide the most information gain.



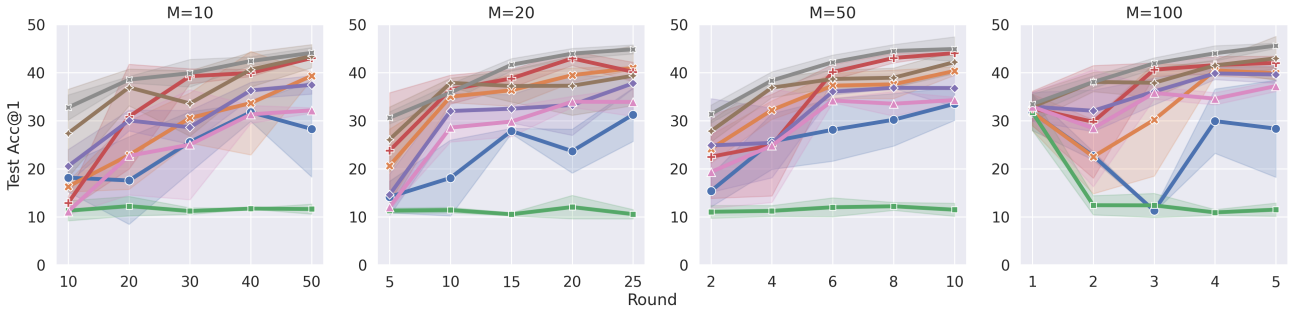
(a) CIFAR10



(b) CIFAR100



(c) Fashion MNIST



(d) CIFAR10 (logistic regression)

Figure 5: Per-round performance comparison (both of the fine-tuning and logistic regression) on three benchmark classification datasets: CIFAR10, CIFAR100, and Fashion MNIST. All of the above sub-figures share the same legend in Fig. 5a. The solid lines are the average result over 3 runs, and the shadow area indicates the standard deviation.

Dataset(↓)	Methods(↓)	M=10				M=20			
		Entropy	CoreSet	BADGE	Random	Entropy	CoreSet	BADGE	Random
CIFAR10	Orig.	52.26	52.84	53.04	56.06	51.88	52.61	53.26	54.43
	w/ FBR+BO	51.02	51.98	54.26	55.25	54.32	50.92	56.98	54.86
	w/ FBR+GS	53.68	54.31	56.90	57.63	56.00	56.22	56.37	56.42
	w/ CHAIN	57.65	59.56	58.03	60.15	55.90	56.90	56.94	57.47
CIFAR100	Orig.	15.52	14.63	16.50	15.63	15.55	15.01	15.77	15.79
	w/ FBR+BO	14.77	14.32	15.92	16.73	14.39	15.44	16.21	16.20
	w/ FBR+GS	16.01	15.59	15.65	16.17	15.34	15.94	16.22	16.17
	w/ CHAIN	17.13	16.12	17.82	18.08	16.62	16.36	16.21	18.82
Fashion MNIST	Orig.	88.50	89.44	87.47	90.08	90.72	89.93	88.64	90.04
	w/ FBR+BO	88.66	86.99	90.30	91.81	88.63	89.46	90.15	90.45
	w/ FBR+GS	89.72	90.54	91.16	89.46	90.44	90.85	91.50	90.06
	w/ CHAIN	91.08	93.41	92.22	91.28	90.99	91.36	91.64	91.48
Dataset(↓)	Methods(↓)	M=50				M=100			
		Entropy	CoreSet	BADGE	Random	Entropy	CoreSet	BADGE	Random
CIFAR10	Orig.	53.57	53.51	54.23	52.25	52.26	53.58	52.12	54.01
	w/ FBR+BO	53.55	53.77	54.20	54.88	51.70	54.65	54.85	54.39
	w/ FBR+GS	54.89	56.65	56.59	55.24	54.63	56.77	56.36	55.89
	w/ CHAIN	58.49	57.14	58.39	58.63	57.48	57.37	58.56	58.74
CIFAR100	Orig.	14.52	15.28	16.38	16.63	14.49	15.68	16.76	15.82
	w/ FBR+BO	13.72	16.13	16.58	16.46	14.04	15.93	14.92	16.36
	w/ FBR+GS	15.37	16.33	16.58	16.57	14.22	16.39	16.36	16.51
	w/ CHAIN	15.95	17.25	17.58	16.83	14.69	16.05	17.51	17.70
Fashion MNIST	Orig.	89.20	88.32	91.88	90.43	90.32	90.13	90.11	89.68
	w/ FBR+BO	90.32	89.59	89.87	90.76	89.77	90.55	89.39	90.89
	w/ FBR+GS	90.44	91.25	91.14	91.56	90.11	91.73	90.66	90.35
	w/ CHAIN	92.12	90.94	92.78	90.90	90.09	90.65	93.47	91.16

Table 3: Final round performance comparison with and without CHAIN of fine-tunning ResNet18 on CIFAR10, CIFAR100 and Fashion MNIST. Blue denotes the best result in each M, and red denotes the best results over all query strategies and M in the corresponding dataset.

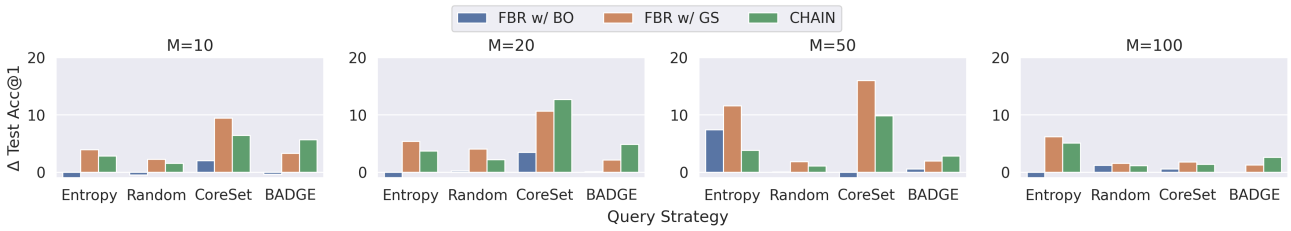
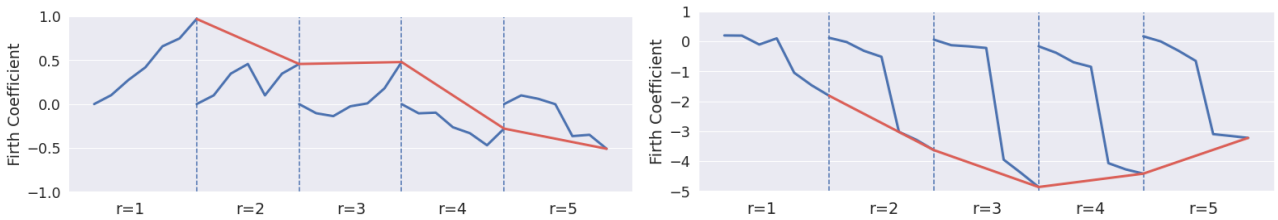


Figure 6: Comparison of the performance gain over all rounds between FBR+BO, FBR+GS and CHAIN of fine-tunning ResNet18 on Fashion MNIST.



(a) Fine-tuning a ResNet-18 on CIFAR100

(b) Fine-tuning a ResNet-18 on Fashion MNIST

Figure 7: The trend of the Firth coefficient across model training steps and AL query rounds on CIFAR100 and Fashion MNIST with random query strategy and $M = 100$.