Journal of Data-centric Machine Learning Research (2024) Submitted 3/24; Revised 4/24; Published 5/24

Learning to Rank for One-Round Active Learning

Zixin Ding ZIXIN@UCHICAGO.EDU
University of Chicago

Si Chen CHENSI@VT.EDU
Virginia Tech

Ruoxi Jia RUOXIJIA@VT.EDU
Virginia Tech

Yuxin Chen CHENYUXIN@UCHICAGO.EDU
University of Chicago

Abstract

Active learning is a promising paradigm to reduce the labeling cost by strategically requesting labels to improve model performance. However, existing active learning methods often rely on expensive acquisition function to compute, extensive modeling retraining and multiple rounds of interaction with annotators. To address these limitations, we propose a novel approach for active learning, which aims to select batches of unlabeled instances through a learned surrogate model for data acquisition. A key challenge in this approach is developing an acquisition function that generalizes well, as the history of data, which forms part of the utility function's input, grows over time. Our novel algorithmic contribution is a multi-task bilevel optimization framework that predicts the relative utility, measured by the validation accuracy, of different training sets, and ensures the learned acquisition function generalizes effectively. For cases where validation accuracy is expensive to evaluate, we introduce efficient interpolation-based surrogate models to estimate the utility function, reducing the evaluation cost. We demonstrate the performance of our approach through extensive experiments on standard active classification benchmarks.

Keywords: Active Learning, Utility Model, Acquisition for ML

1 Introduction

Many decision making tasks involve maximization of utility functions (Chen et al., 2015b; Jackson et al., 2019). As an example, utility in active learning (AL) can be represented in various forms, such as expected error rate reduction (Mussmann et al., 2022; Roy and McCallum, 2001), mutual information between the labeled and unlabeled datasets (Sourati et al., 2016; Adaimi and Thomaz, 2019; Lindley, 1956), or the uncertainty of model predictions (Settles, 2012; Shen et al., 2017; Kossen et al., 2022). However, maximizing utility under budget constraints in AL is notoriously challenging. It is well-known that determining the optimal set containing maximal information under cardinality constraint is NP-hard (Ko et al., 1995; Chen et al., 2015a). In classification tasks, determining the groundtruth utility of *subset* of training data needs retraining classifier on that set (and then evaluate it on the validation set). It's computationally infeasible to calculate out the utility for the *best* possible subset for downstream tasks without carefully examining every possible subset (Engstrom et al., 2024). Moreover, common AL methods rely on acquisition functions with high *adaptivity* to the environment, in which the selection choices for current round depend on the responses

to the labeling requests for all previous rounds. This reliance poses major concerns for the deployment of these algorithms to real-world applications, as there could be a substantial delay between requesting labels and receiving feedback. For instance, in scientific experiments, feedback from wet-lab or physics experiments can take days or even months to obtain (Botu and Ramprasad, 2015; Yang et al., 2019), limiting the rounds of interactions with labelers, thus bearing the risk of sampling redundant or less effective training examples within a batch.

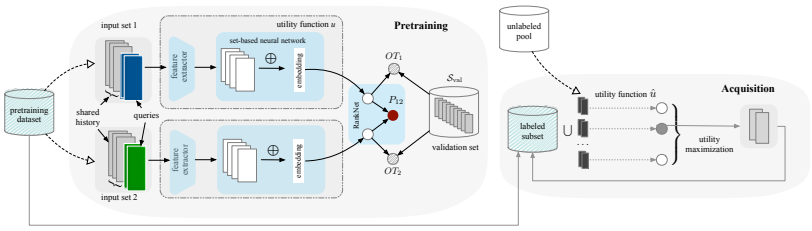


Figure 1: Overview of the RAMBO algorithm. In the pretraining stage, we learn a RankNet over pairs of utility samples via multi-task bilevel optimization; in the acquisition stage, we follow the learned utility function to iteratively query data points in minibatches. Details of the algorithm are provided in Section 3.

In this paper, we focus on enhancing the *robustness* and *generalizability* of deep active learning under one round setting. We propose a novel learning-based acquisition strategy called RAMBO (Ranking-based Active learning via Multitask Bilevel Optimization) that addresses the limitations of existing methods reliant on expensive acquisition functions or overly generic heuristics. Our algorithm is summarized in Fig. 1. Given the variability in deep learning models due to different initializations, hyperparameters, network architectures and training procedures (Jiang et al., 2021; D'Amour et al., 2022; Zhong et al., 2021), the one-shot estimate of validation accuracy can be highly stochastic, and thus we resort to the idea of *ranking* as a strategy to mitigate the inherent uncertainty. To make our approach account for growing labeled pool, we separate samples based on the size of inputs and employ bilevel training to account for the growing training history. We introduce a multi-task learning framework that uses the optimal transport distance (Alvarez-Melis and Fusi, 2020) between the current labeled data and validation set as a supplementary loss, regularizing the utility model to further generalize to unseen unlabeled data, while being oblivious to training dynamics of the underlying classifier. Our experimental results on various image classification tasks demonstrate the effectiveness of the proposed approach.

2 Problem Statement

Consider a ground set of data points $\mathcal{X}$ with a ground truth labeling function $f^* : \mathcal{X} \to \mathcal{Y}$. The active learning problem in our study unfolds in a two-stage protocol: a *pretraining stage* and an *acquisition stage*. Given an initial pool of data points in the pretraining stage, we proceed to actively select a set of new examples to label all at once for the acquisition stage. We denote the initial pretraining (labeled) set by $\mathcal{S}_0$ with $\mathcal{S}_0 \subseteq \mathcal{X}$ and $|\mathcal{S}_0| = k$, and denote the labeled set after the acquisition stage by $\mathcal{S}_1$ with $|\mathcal{S}_1| = k + B$ where B represents the labeling budget. The unlabeled set at initiation and after acquisition is represented as $\mathcal{U}_0$ and $\mathcal{U}_1$ respectively. The groundtruth utility function is defined as $u : 2^{\mathcal{X}} \to \mathbb{R}$, where $u(\xi)$ quantifies the utility of a subset $\xi \subseteq \mathcal{X}$ by evaluating the validation accuracy of the classifier

f induced by the (labeled) data in ξ. Our goal is to find the optimal subset $\mathcal{S}_1^*$ such that f, trained on it achieves maximal validation accuracy, i.e., optimizes the utility function u:

$$\mathcal{S}_1^* \in \underset{\mathcal{S}_0 \subseteq \mathcal{S}_1 \subseteq \mathcal{X}, |\mathcal{S}_1 \setminus \mathcal{S}_0| = B}{\arg\max} u(\mathcal{S}_1) \tag{1}$$

Here, $u(\mathcal{S}_1) = \mathbb{E}_x[\mathbb{1}(f(x) \neq f^*(x))]$ for classification tasks, and can be estimated by the error rate of the resulting f on a validation set $\mathcal{S}_{\text{val}} \subseteq \mathcal{X}$.

In our setting, learning u in Equation 1 is difficult. Indeed, even approximating u requires the groundtruth utility for certain number of susbets of labeled pool, under the practical constraints of limited labeling budget. We emphasize that the instances are selected *non-adaptively* in the acquisition stage, i.e., our selection of instances do not depend on the labels of previous selected instances in the acquisition process. Our goal is to devise an acquisition strategy to perform subset selection which maximally improves downstream classification.

3 Methodology

We introduce our algorithm, RAMBO, following the two-stage learning protocol described previously. In a nutshell, RAMBO (1) collects training samples for *pretraining* utility model, and (2) greedily selects the batch with the maximal estimated utility value from one to total batches t in the acquisition stage. We divide the pretraining stage into τ_1 iterations and acquisition stage into τ_2 iterations with mini-batch size b for each iteration. More precisely, we instantiate RAMBO into following building blocks: a) Develop a set-based multitask neural network model $\hat{u}$ as surrogate model for pretraining; b) define the loss function for the utility model $\hat{u}$; c) sample a collection of subsets $\{(\xi, u(\xi))\}_i \subseteq \mathcal{S}_0$ where $i \in [1, \tau_1]$ as a growing labeled set up to $\mathcal{S}_0$ for training $\hat{u}$; d) update the set based model $\hat{u}$ per iteration of the pretraining stage; e) greedily follow the learned utility model $\hat{u}$ in the acquisition stage.

3.1 A Two-Stage Active Learning Framework

So far, we have defined the framework and will unravel a)-d) above and discuss each relevant aspect respectively:

a) What surrogate models $\hat{u}$ should we use? Similar to Ilyas et al. (2022), by parametrizing a surrogate model with training samples, we transform the surrogate model construction into supervised learning task (See Definition 1). In our context, the training samples ξ are subsets of pretraining set $\mathcal{S}_0$ and the utility value is $u(\xi)$. Throughout this work, we refer the pairs $(\xi, u(\xi))$ as *utility samples*. It is appealing to adopt their linearity assumption into AL setting due to strong theoretical footing (Saunshi et al., 2022) and simplicity in model architectures. Nevertheless, to avoid extensive samples collection and model retraining as Ilyas et al. (2022), we hence prefer more complex architectures for modeling interaction between elements within each utility sample. One natural candidate for $\hat{u}$ is set-based neural networks due to their strong expressive power (i.e., Set Transformer (Lee et al., 2019) or Deep Sets (Zaheer et al., 2017)). Denote the general set-based neural network(NN) as $\text{net}(\xi) = \text{net}(x_1, ..., x_a) = \rho(\text{pool}(\{\phi(x_1), ...\phi(x_a)\})$ where $\{x_i\}_{i=1}^a$ represents a single utility sample ξ with size a and ϕ, ρ is the feature extractor and regressor for the set-based NN itself.

In experiments, we find set-based NN could serve as a primitive for utility model, but still, it lacks principled supervision signals for model training. Engstrom et al. (2024) train millions

of cheap datamodels (Ilyas et al., 2022) in the hope for better generalization for unseen tasks, while in our setting, we cannot afford large-scale training due to computational infeasibility and aim to obtain good utility model with hundreds of samples for faster deployment. Therefore, we need more fine-grained signal that would tie labeled data and validation set *in principle*. In particular, Alvarez-Melis and Fusi (2020) introduce the notion of geometric distance via optimal transport (OT) between two datasets and Just et al. (2023) extend it as a learning-agnostic proxy for measuring model performance on $\mathcal{S}_{val}$. The celebrated success of OT distance in predicting validation set accuracy (Just et al., 2023) enables us to cast the groundtruth OT distance between utility samples and validation set (Alvarez-Melis and Fusi, 2020) as a supervision signal for utility model.[1]

Definition 1 (Surrogate Utility Model) *Let $\mathcal{X}$ be the instance domain, and ξ be any sampled subset drawn from distribution $\mathcal{D}$ over $\mathcal{X}$. A surrogate utility model $\hat{u}(\xi)$ is a set function: $2^{\mathcal{X}} \to \mathbb{R}$, optimized to predict the true utility $u(\xi)$ on a training set $\xi \sim \mathcal{D}$:*

$$\hat{u} = \arg\min_{\tilde{u}_w} \hat{\mathbb{E}}_{\xi \sim \mathcal{D}}[\mathcal{L}(\tilde{u}_w(\xi), u(\xi)] \tag{2}$$

where $\mathcal{L}(\cdot, \cdot)$ denotes the loss function, and $\tilde{u}_w$ is a parametric set function to approximate u.

b) What loss function should we minimize? One natural choice is to directly minimize the MSE(mean square error) of estimated and true utility value as $\mathcal{L} = (\hat{u} - u)^2$. However, the evaluation of validation accuracy is non-deterministic (thus stochastic) due to the aleatoric uncertainty of the classifier itself (Park et al., 2023). While the simplistic way is to train a neural network to approximate the utility value in regression fashion and minimize the MSE, we fail to learn a good utility model by regressing validation accuracy on set of utility samples (See Section B.2.2 for ablation study on casting utility model as regression). A natural way to design our loss function lies in the idea of pairwise ranking, simplifying regression problem to ranking problem. Yoo and Kweon (2019) introduce a loss prediction module to predict the classifier loss on single data point and handicraft the loss function for predicting the classifier loss in pairwise ranking fashion. For a minibatch samples with size d, Yoo and Kweon (2019) divide it into $d/2$ pairs and rank the differences between each pair of predicted and groundtruth losses to discard the overall loss scale. Extending the idea of ranking classification loss between pairs of data point to rank the utility value, we incorporate the classical RankNet (Burges et al., 2005) structure to rank between pairs of equal size utility samples with OT distance as a regularizer in the final loss.

Definition 2 (Ranking Loss) *Given $\mathcal{X}$, and let ξ_1, ξ_2 be two sampled subset drawn from distribution $\mathcal{D}$ over $\mathcal{X}$ with equal size d. Denote the utility value (validation accuracy) of ξ_1 as u_1 and the utility value of ξ_2 as u_2. W.l.o.g. suppose $u_1 > u_2$, $u_{12} = u_1 - u_2$. Specifically, $u_1 > u_2$ is taken to mean that the surrogate utility model $\hat{u}$ asserts that $\xi_1 \rhd \xi_2$. Denote the modeled posterior $P(u_1 \rhd u_2)$ by P_{12}, and let $\bar{P}_{12}$ be the desired target values for those posteriors. The Binary Cross Entropy(BCE) loss for pair (ξ_1, ξ_2) is written as*

$$\mathcal{L}_{Rank_{12}} = -\bar{P}_{12} \log P_{12} - (1 - \bar{P}_{12}) \log(1 - P_{12}).$$

1. For implementation of OT distance calculation, we use https://github.com/microsoft/otdd between each utility sample and validation dataset.

Definition 3 (OT Distance Loss) *Given two utility samples ξ_1, ξ_2 and its corresponding ground truth OT distance value as OT_1, OT_2 and the predicted values as $\hat{OT}_1$ and $\hat{OT}_2$. The loss is defined as $\mathcal{L}_{OT} = \lambda_1(\hat{OT}_1 - OT_1)^2 + \lambda_2(\hat{OT}_2 - OT_2)^2 - \lambda_3(\min(\hat{OT}_1, 0) + \min(\hat{OT}_2, 0))$ where $\lambda_1, \lambda_2, \lambda_3$ are hyperparameters. The first two terms are mean squared error for OT distances and the third terms are positive constraints.*

Definition 4 (Total Loss for Utility Model) $\mathcal{L}_{Total} = \mathcal{L}_{Rank_{12}} + \lambda_{OT} \cdot \mathcal{L}_{OT}$ *where λ_{OT} is a hyperparameter.*

c) How do we collect utility samples iteratively? The very first question encountered during pretraining is how to generate utility samples. Ilyas et al. (2022) construct training subsets by random sampling a fixed-length subset. One caveat in our setting is the growing length of labeled set as the progression of the active learner. To enable the model to adapt to growing length of utility samples, one needs to incorporate *diversity* in the size of ξ. One natural choice is to perform rejection sampling from the *powerset* of $\mathcal{S}_0$, i.e., $\xi \sim 2^{\mathcal{S}_0}$. Instead of fixing the sampling proportion, we propose to fix the number of utility samples collected from $\mathcal{S}_0$ per iteration during pretraining as n.

d) How do we update the set-based NN during pretraining? As mentioned in Section 3.1, the length of labeled utility samples grows and random split for training and validation set may fail to capture the notion of generalizability in neural batch active learning. The goal of utility model is to *generalize* to longer length of utility samples and learn a general mapping from utility sample to validation accuracy. Inspired by bilevel training work (Franceschi et al., 2018; Grazzi et al., 2020; Borsos et al., 2021), we employ a bilevel framework to separate the utiltiy samples by *length*. In practice, we separate out the validation set and training set by 50% and 50% for simplicity. We retrain the set-based NN per iteration with the accumulation of utility samples per iteration. We defer the complete discussion of bi-level training to Section 3.2.

e) How do we acquire data in the acquisition stage? In the context of utility maximization, perhaps the simplest candidate is to select the instance with largest predicted utility. Popular approaches rely on sequentially picking one data point per round (Houlsby et al., 2011; Gal et al., 2017) though the addition of single data point cause minimal change to validation accuracy while increasing the cost of model retraining. Alieva et al. (2020) suggest that for many sequential decision making problems, greedy heuristics for sequentially selecting actions exhibits superior performance without invoking expensive evaluation oracles. Recall that one could interpret $\hat{u}$ as a score-based acquisition function and leverage it for sequential decision making, i.e. to greedily select unlabeled data with highest predicted utility. Inspired by Citovsky et al. (2021), we employ Margin Sampling (Roth and Small, 2006) as a filter for unlabeled instance i.e., select M unlabeled instances with lowest margin scores, per iteration in the acquisition stage (See Fig. 2). We propose to randomly split $\mathcal{U}_0$ into batches of size b, concatenate each batch to the current labeled pool, and then use the concatenated batch as input to $\hat{u}$ for utility prediction. We perform sequential batch selection within the acquisition stage and select the unlabeled batch with the largest predicted score.

3.2 The RAMBO Algorithm

Bi-Level Optimization We initialize the utility model by collecting and training samples from offline datasets, providing an initial estimate of the *feature extractor* ϕ_0. To align with

Input: $B, \mathcal{U}_0, \mathcal{S}_0\ \mathcal{X}, b, M, n, \mathcal{S}_{val}$.
Output: $\mathcal{S}_1$
Initialize $(\hat{u}_0, \phi_0)$ from offline dataset
Randomly divide $\mathcal{S}_0$ into S_0 with size k_1 and $\{s_1, s_2 ... s_{\tau_1}\}$ with each size b
Set $U_0 = \mathcal{U}_0$, $\tau_1 = \frac{k-k_1}{b}$ and $\tau_2 = \frac{B}{b}$
Train f on S_0 and get accuracy on $\mathcal{S}_{val}$ as acc_0
$\mathcal{D}_0 \leftarrow \{\}$
for $i = 0 : \tau_1$ **do** ▷ **Pretraining**
    $S_{i+1} \leftarrow S_i \cup \{s_{i+1}\}$
    Train f on S_{i+1}
    Obtain accuracy on $\mathcal{S}_{val}$ as acc_{i+1}
    $D_{i+1} \leftarrow$
Utility-Samples-Augmentation$(S_i, S_{i+1}, n, acc_i, acc_{i+1}, D_i)$
    Train $\hat{u}_i$ from D_{i+1} ▷ **Bilevel Optimization**
for $j = 0 : \tau_2$ **do** ▷ **Acquisition**
    $S_{j+1}, U_{j+1} \leftarrow$
Greedy-Margin$(\hat{u}_{\tau_1}, j, b, S_j, M, U_j)$
$\mathcal{S}_1, \mathcal{U}_1 = S_{\tau_2}, U_{\tau_2}$

procedure Greedy-Margin
    Input: $\hat{u}$, j, b, S_j, M, U_j.
    Output: S_{j+1}, U_{j+1}
    $R \rightarrow$ a subset obtained by smallest margin scores M examples from $U_j \setminus S_j$
    Randomly divide R into $\{\lfloor \frac{R}{b} \rfloor\}$ batches of subsets $\{(x_i)_{i=1}^b\}$.
    $b_{\max} \leftarrow \arg\max_{\{(x_i)_{i=1}^b\} \in \{\lfloor \frac{R}{b} \rfloor\}} \hat{u}(S_j \cup (x_i)_{i=1}^b)$
    $S_{j+1} \leftarrow S_j \cup \{b_{\max}\}$
    $U_{j+1} \leftarrow U_j \setminus \{b_{\max}\}$

procedure Utility-Samples-Augmentation
    Input: S_i, S_{i+1}, n, acc_i, acc_{i+1}, D_i.
    Output: D_i
    for $i \in$ range(n) **do**
        Sample (ξ_1, ξ_2) from S_i with equal size
        Compute distance between $\phi(\xi_1)$ and $\phi(S_i)$ as $d_{1,i}$ and distance between $\phi(\xi_1)$ and $\phi(S_{i+1})$ as $d_{1,i+1}$. Same Rule applies to ξ_2 to obtain $d_{2,i}$ and $d_{2,i+1}$.
        Calculate u_1, u_2 for ξ_1 and ξ_2 by Definition 5
        $D_i \leftarrow D_i \cup \{(\xi_1, u_1), (\xi_2, u_2)\}$

Figure 2: The RAMBO Algorithm

growing labeled pool of active learning setting, a core requirement of our utility model is the capability to *generalize to longer and unseen data* by drawing on prior utility samples. A line of research (Rajeswaran et al., 2019; Liu et al., 2019) suggests that meta-learning shall lead to fast adaptation and generalization to new tasks. One formulation of meta-learning is bi-level optimization (Maclaurin et al., 2015) where the inner objective represents adaptation to a given task and the outer problem is the meta-training objective. Motivated by Franceschi et al. (2018), we formulate utility model training as bilevel optimization, combining gradient-based hyperparameter optimization and meta-learning in which the outer optimization problem is solved subject to the optimality of an inner optimization problem. To improve the utility model's generalization capability on samples with varied lengths, we divide the utility samples $(\xi, u(\xi))$ at iteration i to training D_{tr} and validation set D_{val} by length, where D_{tr} corresponds to utility samples with length smaller than the median and vice versa, and treat them as input dataset for *inner objective* L and *outer objective* E. Formally, we consider the bilevel optimization framework as $\min_\lambda\ E(w(\lambda), \lambda)$ s.t. $w(\lambda) = \arg\min_{\hat{w} \in \mathbb{R}^d} \mathcal{L}(\hat{w})$ where λ is a hyperparameter, E and $\mathcal{L}$ are continuously differentiable functions, the outer objective $E(w(\lambda), \lambda) := \sum_{\{(S'_1, u(S'_1)), (S'_2, u(S'_2))\} \in D_{val}} \mathcal{L}_{\text{Total}}(\hat{w})$, and the inner objective as $\mathcal{L}(\hat{w}) = \sum_{\{(S'_1, u(S'_1)), (S'_2, u(S'_2))\} \in D_{tr}} \mathcal{L}_{\text{Total}}(\hat{w}) + \Omega_\lambda(\hat{w})$ where $D_{\text{tr}} = \{(\xi_1, u(\xi_1)), (\xi_2, u(\xi_2))\}_{i=1}^n$ is a set of pair of utility samples attributed to training set and $\mathcal{L}_{\text{Total}}(\cdot)$ is the BCE loss induced by the supervised algorithm and Ω_λ is a regularizer parametrized by λ. The outer objective is the proxy of generalization error of $\hat{u}(\cdot)$, given by the average loss of D_{val}.

The inner optimization is aimed at *utility model optimization*, i.e., finding the best parameters that minimizing the loss on smaller length training samples D_{tr}. Conversely, the outer optimization targets to *generalize the model to longer-length utility samples* D_{val}, which seeks the optimal regularizer parameterized by λ. With this bilevel formulation, RAMBO

shows better and more stable performance when performing unlabeled data selection on CIFAR10 with labeling budge 5000 (as suggested by Table 3). Table 3 shows average performance of models with bilevel training used in optimization, mostly outperforms the rest of counterparts without bilevel training, illustrating the enhanced generalizability across various model architectures and training algorithms.

Interpolation-Based Utility Samples In contrast to thousands or even millions of training samples for datamodels framework Ilyas et al. (2022); Engstrom et al. (2024), the scarcity of utility samples poses challenges to the efficacy of our utility model training. We resort to the consistency regularization techniques from semi-supervised learning to augment artificial $(\xi, u(\xi))$. Inspired by Parvaneh et al. (2022), the latent space of the classifier's feature extractor shall contain valuable representations that can be interpolated within labeled instances. The empirical success suggests a change in perspective—rather than twisting the classifier, we leverage the shared representations in $\hat{u}$ throughout the progress of optimization. In particular, we adopt the *interpolation consistency regularization* strategy (Verma et al., 2022) (Definition 5). The pseudo code for utility samples augmentation is outlined in Fig. 2.

Definition 5 (Utility Value Interpolation) *Denote the validation accuracy at iteration i as acc_i. For a given utility sample ξ_1, let $d_{1,i}$ be its distance with the previous labeled pool S_i and $d_{1,i+1}$ the distance with the current labeled pool S_{i+1}. The augmented utility value u_1 for ξ_1 yields as $u_1 = \alpha \cdot u_i + (1-\alpha) \cdot u_{i+1}$ with $\alpha := \frac{d_{1,i+1}}{d_{1,i+1}+d_{1,i}}$.*

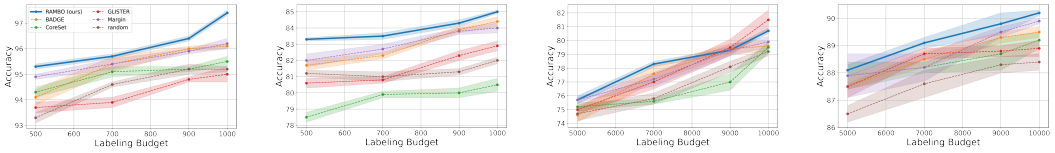
4 Experimental Results

Experimental Setup Here, we evaluate the performance of RAMBO against several state-of-the-art baselines on four image datasets MNIST (LeCun et al., 1998), FashionMNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky et al., 2009), SVHN(Netzer et al., 2011). To ensure a comprehensive comparison among all algorithms, we evaluate them across various acquisition stage budget B as {500, 700, 900, 1000} for MNIST and FashionMNIST with $k = 200$, {5000, 7000, 9000, 10000} for CIFAR10 and SVHN with $k = 2500$. For main results, we focus on the accuracy of validation set as the key performance metric. We fix the validation size to be 1000 across all datasets. Lastly, we run each experiment ten times and report average and standard error across all experiments. Depending on the type of dataset, we consider different network architectures for classifiers. We consider two classifier structures: one is Beck et al. (2021)'s neural network structure, a model similar to LeNet (LeCun et al., 1998) for MNIST and FashionMNIST and the other is ResNet-18 (He et al., 2016) for CIFAR10 and SVHN. We provide details of utility model architecture to the Appendix.

We fit all classifiers using cross-entropy loss with optimizer Adam until training accuracy exceeds 99% with maximum 100 epochs and learning rate 0.001. No learning rate schedulers and data augmentations are used. [2]

Baselines For all experiments, we consider the following baselines: **Margin Sampling**: Selects B examples from $\mathcal{U}_0$ with smallest difference between the first and second most

2. Baselines use implementations from open-source AL toolkit DISTIL Team (2023). All models are trained in PyTorch (Paszke et al., 2017).



(a) MNIST (b) FashionMNIST (c) CIFAR10 (d) SVHN

Figure 3: Experimental results: Accuracy vs. Labeling budget. Results are given in %.

probable classes predicted by f (Roth and Small, 2006). **BADGE**: A hyperparameter-free approach that trades off between diversity and uncertainty using k-means++ in hallucinated gradient space (Ash et al., 2019). **CoreSet**: A diversity based approach using greedy approximation to the k-center problem on representations from the current classifier's penultimate layer (Sener and Savarese, 2017). **GLISTER**: A learning-based approach selecting B instances from $\mathcal{S}_0$ that would maximize the log-likelihood on held-out validation set $\mathcal{S}_{val}$ by converting it as a mixed discrete continuous bilevel optimization. We adopt the GLISTER-ONLINE version as an approximation for the inner optimization problem by taking a single gradient step update (Killamsetty et al., 2021). **Random**: Selects B samples uniformly at random.

Results In Figure 3, RAMBO outperforms most of baselines across multiple architectures and various labeling budget for acquisition stage. For easy datasets like FashionMNIST and MNIST, RAMBO shall learn a good shared representation for effective utility value interpolation and can easily beats all the baselines oblivious to different labeling budgets which suggests RAMBO is a good choice regardless of labeling budget. As BADGE and CoreSet operate on the penultimate layer with limited budget, both algorithms fail to perform well as their learned representations might not be accurate. However, RAMBO performs interpolation techniques to augment utility samples within limited labeled pool and generalize to predictions of longer history of labeled data, leading to learning-based acquisition function amenable to growing labeled pool.

Even for hard datasets, for CIFAR10 and SVHN, when the model fails to have a good architecture priors due to limited pool, BADGE and CoreSet cannot learn meaningful representations. Occationally, CoreSet might not outperform passive learning, for instance, labeling Budget is 7000 for CIFAR10.

5 Conclusion

We have demonstrated existing state-of-the-art methods are suboptimal in single round selection. We show that under certain budget for pretraining, RAMBO would achieve better generalization performance compared to other active learning algorithms, and that most of validation accuracy improvement is realized by our two-stage algorithm. Finally, we illustrate how behaviors of all algorithms change with variation of pretraining and single round acquisition budget across multiple datasets and architectures. One potential direction for future work could be to determine an optimal budget allocation for both the pretraining and acquisition stages, as well as the extension of RAMBO to the few-rounds setting.

References

R. Adaimi and E. Thomaz. Leveraging active learning and conditional mutual information to minimize data annotation in human activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(3):1–23, 2019.

A. Alieva, A. Aceves, J. Song, S. Mayo, Y. Yue, and Y. Chen. Learning to make decisions via submodular regularization. In *International Conference on Learning Representations*, 2020.

D. Alvarez-Melis and N. Fusi. Geometric dataset distances via optimal transport. *Advances in Neural Information Processing Systems*, 33:21428–21439, 2020.

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

P. Bachman, A. Sordoni, and A. Trischler. Learning algorithms for active learning. *CoRR*, abs/1708.00088, 2017. URL `http://arxiv.org/abs/1708.00088`.

N. Beck, D. Sivasubramanian, A. Dani, G. Ramakrishnan, and R. Iyer. Effective evaluation of deep active learning on image classification tasks. *arXiv preprint arXiv:2106.15324*, 2021.

Z. Borsos, M. Tagliasacchi, and A. Krause. Semi-supervised batch active learning via bilevel optimization. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3495–3499. IEEE, 2021.

V. Botu and R. Ramprasad. Adaptive machine learning framework to accelerate ab initio molecular dynamics. *International Journal of Quantum Chemistry*, 115(16):1074–1083, 2015.

C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005.

Y. Chen, S. H. Hassani, A. Karbasi, and A. Krause. Sequential information maximization: When is greedy near-optimal? In *Conference on Learning Theory*, pages 338–363. PMLR, 2015a.

Y. Chen, S. Javdani, A. Karbasi, J. Bagnell, S. Srinivasa, and A. Krause. Submodular surrogates for value of information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015b.

G. Citovsky, G. DeSalvo, C. Gentile, L. Karydas, A. Rajagopalan, A. Rostamizadeh, and S. Kumar. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34:11933–11944, 2021.

C. Coleman, C. Yeh, S. Mussmann, B. Mirzasoleiman, P. Bailis, P. Liang, J. Leskovec, and M. Zaharia. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*, 2019.

A. D'Amour, K. Heller, D. Moldovan, B. Adlam, B. Alipanahi, A. Beutel, C. Chen, J. Deaton, J. Eisenstein, M. D. Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *The Journal of Machine Learning Research*, 23(1):10237–10297, 2022.

L. Engstrom, A. Feldmann, and A. Madry. Dsdm: Model-aware dataset selection with datamodels. *arXiv preprint arXiv:2401.12926*, 2024.

M. Fang, Y. Li, and T. Cohn. Learning how to active learn: A deep reinforcement learning approach. *CoRR*, abs/1708.02383, 2017. URL `http://arxiv.org/abs/1708.02383`.

L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pages 1568–1577. PMLR, 2018.

Y. Gal, R. Islam, and Z. Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

R. Grazzi, L. Franceschi, M. Pontil, and S. Salzo. On the iteration complexity of hypergradient computation. 2020.

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

N. Houlsby, F. Huszár, Z. Ghahramani, and M. Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

A. Ilyas, S. M. Park, L. Engstrom, G. Leclerc, and A. Madry. Datamodels: Understanding predictions with data and data with predictions. In *International Conference on Machine Learning*, pages 9525–9587. PMLR, 2022.

C. Jackson, A. Presanis, S. Conti, and D. De Angelis. Value of information: Sensitivity analysis and research design in bayesian evidence synthesis. *Journal of the American Statistical Association*, 114(528):1436–1449, 2019.

Y. Jiang, V. Nagarajan, C. Baek, and J. Z. Kolter. Assessing generalization of sgd via disagreement. In *International Conference on Learning Representations*, 2021.

H. A. Just, F. Kang, T. Wang, Y. Zeng, M. Ko, M. Jin, and R. Jia. LAVA: Data valuation without pre-specified learning algorithms. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=JJuP86nBl4q`.

A. Karatzoglou, L. Baltrunas, and Y. Shi. Learning to rank for recommender systems. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 493–494, 2013.

K. Killamsetty, D. Sivasubramanian, G. Ramakrishnan, and R. Iyer. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8110–8118, 2021.

C.-W. Ko, J. Lee, and M. Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 43(4):684–691, 1995.

K. Konyushkova, R. Sznitman, and P. Fua. Learning active learning from data. *Advances in neural information processing systems*, 30, 2017.

J. Kossen, S. Farquhar, Y. Gal, and T. Rainforth. Active surrogate estimators: An active learning approach to label-efficient model evaluation. *Advances in Neural Information Processing Systems*, 35:24557–24570, 2022.

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

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

H. Li. *Learning to rank for information retrieval and natural language processing*. Springer Nature, 2022.

M. Li, X. Liu, J. van de Weijer, and B. Raducanu. Learning to rank for active learning: A listwise approach. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5587–5594. IEEE, 2021.

Y. Li and J. Oliva. Active feature acquisition with generative surrogate models. In *International Conference on Machine Learning*, pages 6450–6459. PMLR, 2021.

D. V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

S. Liu, A. Davison, and E. Johns. Self-supervised generalisation with meta auxiliary learning. *Advances in Neural Information Processing Systems*, 32, 2019.

T.-Y. Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

D. Maclaurin, D. Duvenaud, and R. Adams. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*, pages 2113–2122. PMLR, 2015.

S. Mussmann, J. Reisler, D. Tsai, E. Mousavi, S. O'Brien, and M. Goldszmidt. Active learning with expected error reduction. *arXiv preprint arXiv:2211.09283*, 2022.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

S. M. Park, K. Georgiev, A. Ilyas, G. Leclerc, and A. Madry. Trak: Attributing model behavior at scale. *arXiv preprint arXiv:2303.14186*, 2023.

A. Parvaneh, E. Abbasnejad, D. Teney, G. R. Haffari, A. Van Den Hengel, and J. Q. Shi. Active learning by feature mixing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12237–12246, 2022.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019.

D. Roth and K. Small. Margin-based active learning for structured output spaces. In *Machine Learning: ECML 2006: 17th European Conference on Machine Learning Berlin, Germany, September 18-22, 2006 Proceedings 17*, pages 413–424. Springer, 2006.

N. Roy and A. McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, 2:441–448, 2001.

N. Saunshi, A. Gupta, M. Braverman, and S. Arora. Understanding influence functions and datamodels via harmonic analysis. *arXiv preprint arXiv:2210.01072*, 2022.

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

B. Settles. Active learning. *Synthesis lectures on artificial intelligence and machine learning*, 6(1):1–114, 2012.

Y. Shen, H. Yun, Z. C. Lipton, Y. Kronrod, and A. Anandkumar. Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*, 2017.

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

J. Sourati, M. Akcakaya, J. G. Dy, T. K. Leen, and D. Erdogmus. Classification active learning based on mutual information. *Entropy*, 18(2):51, 2016.

D. Team. distil. `https://github.com/decile-team/distil`, 2023.

V. Verma, K. Kawaguchi, A. Lamb, J. Kannala, A. Solin, Y. Bengio, and D. Lopez-Paz. Interpolation consistency training for semi-supervised learning. *Neural Networks*, 145: 90–106, 2022.

T. Wang, S. Chen, and R. Jia. One-round active learning. *CoRR*, abs/2104.11843, 2021. URL `https://arxiv.org/abs/2104.11843`.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

B. Xie, L. Yuan, S. Li, C. H. Liu, and X. Cheng. Towards fewer annotations: Active learning via region impurity and prediction uncertainty for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8068–8078, 2022.

S. Yan, S. Zhang, and X. He. Budget-aware few-shot learning via graph convolutional network. *arXiv preprint arXiv:2201.02304*, 2022.

K. K. Yang, Z. Wu, and F. H. Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

O. Yehuda, A. Dekel, G. Hacohen, and D. Weinshall. Active learning through a covering lens. *arXiv preprint arXiv:2205.11320*, 2022.

D. Yoo and I. S. Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 93–102, 2019.

M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

R. Zhong, D. Ghosh, D. Klein, and J. Steinhardt. Are larger pretrained language models uniformly better? comparing performance at the instance level. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3813–3827, 2021.

Appendix A. Related Work

Utility model learning Surrogate models build on a rich and growing body of machine learning literature (Konyushkova et al., 2017; Coleman et al., 2019; Kossen et al., 2022; Ilyas et al., 2022; Engstrom et al., 2024). These works improve data acquisition by: training a regressor to predict expected error rate reduction (Konyushkova et al., 2017), trimming down model architectures/training epochs as proxy models (Coleman et al., 2019), approximate the distribution of labels and unobserved features (Li and Oliva, 2021), and harness datamodels (Ilyas et al., 2022) framework by minimizing trained model loss on target tasks (Engstrom et al., 2024). In contrast, our method use ranking-based neural networks to *learn* a data acquisition function(utility model) *estimating* which subset would yield higher utility value given a pair of equal size of subset training data. Contrary to Ilyas et al. (2022), we train the utility model by collecting much less samples and sampling from various sizes of subsets rather than fixing subset sizes.

Planning-based vs learning-based AL strategies Classical AL have predefined acquisition strategy including uncertainty sampling (Settles, 2012; Shen et al., 2017; Gal et al., 2017), diversity sampling (Sener and Savarese, 2017; Yehuda et al., 2022) or their combined approaches (Xie et al., 2022; Citovsky et al., 2021). Meanwhile, there is a long line of work on *learning-based* acquisition function (Fang et al., 2017; Bachman et al., 2017; Wang et al., 2021; Sinha et al., 2019; Yan et al., 2022; Yoo and Kweon, 2019; Li and Oliva, 2021; Killamsetty et al., 2021). For instance, Killamsetty et al. (2021) cast designing acquisition function into bi-level optimization framework and jointly optimizing model parameters for both training and validation loss. Borsos et al. (2021) extend bi-level optimization to design learning-based acquisition function in the fashion of semi-supervised learning. Yoo and Kweon (2019) adopt the idea of ranking the predicted classifier loss in comparing two instances as "loss prediction module", querying instances that the classifier is likely to predict wrong, and learn it to predict target losses of unlabeled inputs. We draw inspirations from Killamsetty et al. (2021); Yoo and Kweon (2019) by leveraging bi-level training as a subroutine for learning generalizable utility model incorporating growing labeled pool and reducing utility maximization problem by choosing highest ranked batch of unlabeled instances.

Learning to rank Ranking techniques have been foundational in fields such as information retrieval (Liu et al., 2009), recommendation systems (Karatzoglou et al., 2013; Li, 2022) and large language models (Ouyang et al., 2022). Motivated by Yoo and Kweon (2019); Li et al. (2021), we shift from the traditional approach of learning cross-entropy loss on unlabeled instances to ranking the utility for paired subsets of data. While both works (Yoo and Kweon, 2019; Li et al., 2021) view ranking predicted losses as an uncertainty measure, our methodology centers on gauging the utility of labeled data subsets, with the utility being the validation accuracy post-training. To the best of our knowledge, our method is the first to incorporate the idea of ranking between pairs of subsets and link it directly to performance of the learning algorithm on the validation set. Our unique contribution lies in introducing such a ranking mechanism and integrating it under the RankNet (Burges et al., 2005) framework, in tandem with plugging in optimal transport distance under a multitask learning scenario.

Appendix B. Supplemental Experimental Results

B.1 Utility Model Architecture

Here, we describe the acquisition network (Utility Model) discussed in Section 4.

B.1.1 MNIST and FashionMNIST

Our architecture performs the following operations on pairs of subsets of images (Utility Samples) with equal size. We use below networks as feature extractor for pairs of raw embeddings of images. For each one within the pair:

1. 2-D convolution on set of images.
2. 2-D Max Pool on output of (1).
3. ReLU on output of (2).
4. 2-D DropOut on output of (3).
5. 2-D Max Pool on output of (4).
6. ReLU on output of (5).
7. Fully-Connected Layer on Output of (6).
8. ReLU on output of (7).
9. 2-D DropOut on output of (8).
10. Fully-Connected Layer on output of (9).
11. ReLU on output of (10).

B.1.2 CIFAR10 and SVHN

We use pretrained ResNet-18 on ImageNet as feature extractor and perform the following operations on pairs of subsets of extracted features for each image. For each one within the pair:

1. Fully Connected Layer on set of feature embeddings.
2. ReLU on output of (1).
3. Fully Connected Layer on output of (2).

B.1.3 Mutitask Set-based Neural Networks with RankNet

After average pooling of output of (11) for MNIST and FashionMNIST and output of (3) for CIFAR10 and SVHN, for each one within the pair, we perform the following operations:

1. Fully Connected Layer on extracted features
2. ReLU on output of (1).
3. Fully Connected Layer on output of (2).
4. Sigmoid function on output of (3).

Denote the output of (4) as ϕ_1 and ϕ_2.

For the prediction of probability score that which subset has larger utility value in the pair, we apply RankNet on ϕ_1 and ϕ_2 for pair comparison. The output score predicted by RankNet is the final probability score that we shall use to determine whether the first set has larger utility value than the second.

For the interpolation of utility value, we use ϕ_1 and ϕ_2 as embedding.

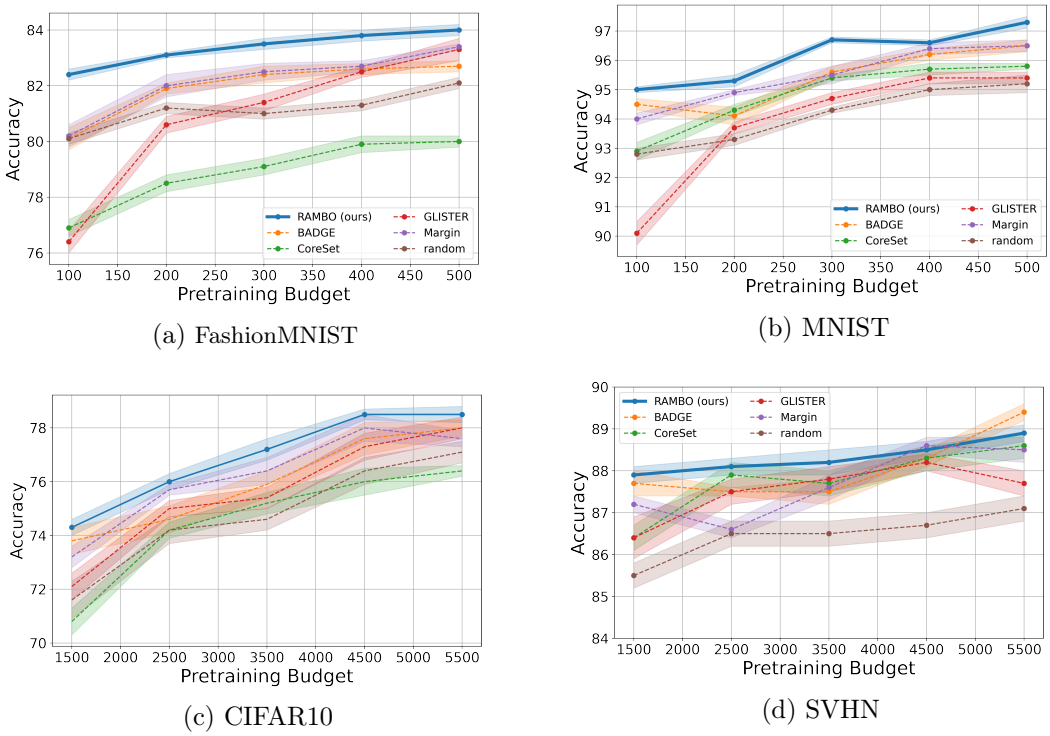
For the prediction of optimal transport distance, we use MLP projection head for ϕ_1 and ϕ_2:

1. Fully Connected Layer on ϕ_1 and ϕ_2
2. ReLU on outputs of (1)
3. Fully-Connected Layer on outputs of (2).

We use the outputs of (3) as a supervision signal in designing the loss function for the neural acquisition function (see Definition 3 in Section 3).

We chose λ_1, λ_2 to be 0.5 and λ_3 to be 1.

B.2 Supplemental Experimental Results



(a) FashionMNIST

(b) MNIST

(c) CIFAR10

(d) SVHN

Figure 4: Experimental results: Ablation on k.

B.2.1 Size of Pretraining Set

Figure 4 illustrates the impact of the size of pretraining set on final validation set accuracy. One shall see RAMBO outperforms the rest of baselines with most of pretraining splits. The only outlier case could be SVHN, similar to CIFAR10 setting. One possibility could be $k = 5500$ is suffice for BADGE to learn an accurate-enough gradient embedding space for single round selection. Another interesting observation is GLISTER often performs worse than most of baselines for FashionMNIST and MNIST when pretraining budget is extremely low as $k = 100$. One possible explanation could be extremely small pretraining budget can not guarantee good inner-level optimization for maximizing training set log-likelihood for extremely small labeled data.

B.2.2 Bi-level training, OT Distance and RankNet

Next, we shift to study the intertwined effects of three design choices. To prove the efficacy of synergizing three seemingly irrelevant submodules together, we provide ablation study of three submodules. Table 1, 2, 3 and 4 show the impact of turning off each submodule on the final validation set accuracy for FashionMNIST, MNIST, CIFAR10 and SVHN respectively.

The cross mark for RankNet means regression based acquisition function and the loss is designed as MSE between predicted utility vs. true utility value. One thing to note is that if the performance of regression based acquisition function without bi-level training and OT distance is similar to random, which corroborates our intuition about ranking instead of regressing validation accuracy on labeled samples.

For simplicity, the checkmarks for optimal transport means $\lambda_{OT} = 1$ and the crossmarks for RankNet denotes regression-based utility model as stated in the main paper. In particular, we only collect single utility sample and develop multitask learning framework on the single utility sample. We still use the feature extractor explained in Section B.1.1 for MNIST and FashionMNIST and Section B.1.2 for CIFAR10 and SVHN. For the regression style acquisition function, we impose MLP head on the shared representation space ϕ for predicting validation accuracy with $\hat{u} = g(\phi) = W^{(2)}(\sigma(W^{(1)}))$ where σ is a RELU non-linearity, very much similar to the description of predicting OT distance in Section B.1.3. For OT distance regularization, we adopt the same MLP projection head architecture described in Section B.1.3.

Table 1: Ablation study on three submodules with pretraining set $k = 200$ and acquisition budget $B = 500$ for FashionMNIST. The last row corresponds to the random baseline.

Bilevel	Optimal Transport	RankNet	Accuracy
✓	✓	✓	$\mathbf{83.1 \pm 0.1}$
✓	✓	×	81.9 ± 0.2
✓	×	✓	81.2 ± 0.4
✓	×	×	81.8 ± 0.2
×	✓	✓	81.0 ± 0.3
×	✓	×	81.7 ± 0.2
×	×	✓	80.9 ± 0.3
×	×	×	81.6 ± 0.1
-	-	-	81.2 ± 0.2

B.2.3 Hyperparameter Tuning for OT distance

By definition, $\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{Rank}_{12}} + \lambda_{\text{OT}} \cdot \mathcal{L}_{\text{OT}}$ (Definition 4). One can change the scale of λ_{OT} for utility model training in pretraining. We study the effect of hyperparameter λ_{OT} in final model performance on validation set. We highlight the importance of incorporating OT distance into the loss structure which makes $\hat{u}$ insensitive to the scale of λ_{OT}. When $\lambda_{\text{OT}} > 0$, the overall validation accuracy is larger than $\lambda_{\text{OT}} = 0$. The choice of λ_{OT} is specific to dataset and batch setting and we present one setting of λ_{OT} with varied Labeling Budget for acquisition stage in Figure 5c.

Figure 5 illustrates the benefits of incorporating optimal transport distance into the loss structure of our utility model. Figure 5 shall serve as a complement to uncover the usefulness

Table 2: Ablation study on three submodules with pretraining set $k = 200$ and acquisition budget $B = 500$ for MNIST. The last row corresponds to the random baseline.

Bilevel	Optimal Transport	RankNet	Accuracy
✓	✓	✓	$\mathbf{95.3 \pm 0.2}$
✓	✓	×	94.9 ± 0.2
✓	×	✓	95.0 ± 0.1
✓	×	×	94.8 ± 0.2
×	✓	✓	94.6 ± 0.1
×	✓	×	94.9 ± 0.1
×	×	✓	95.0 ± 0.2
×	×	×	94.8 ± 0.2
-	-	-	93.4 ± 0.1

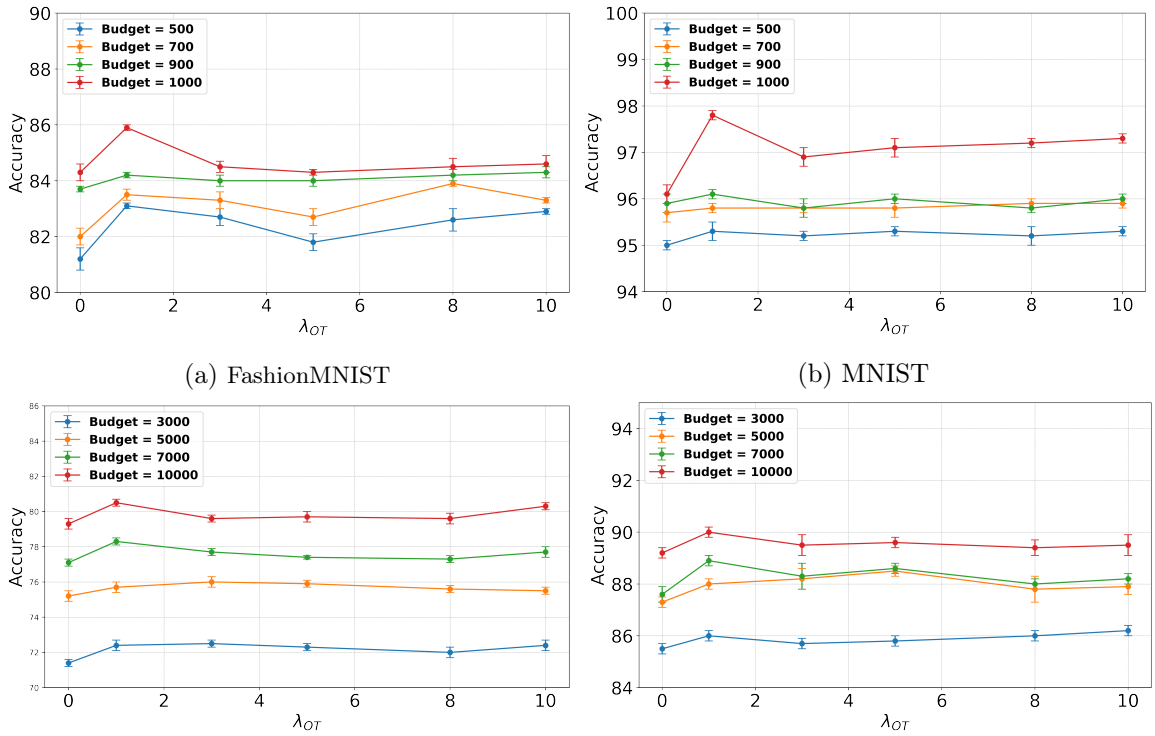
Table 3: Ablation study on three submodules with pretraining set $k = 3500$ and acquisition budget $B = 5000$. The last row corresponds to the random baseline.

Bilevel	Optimal Transport	RankNet	Accuracy
✓	✓	✓	$\mathbf{77.3 \pm 0.2}$
✓	✓	×	76.1 ± 0.3
✓	×	✓	76.2 ± 0.4
✓	×	×	70.5 ± 0.3
×	✓	✓	75.5 ± 0.3
×	✓	×	75.5 ± 0.3
×	×	✓	76.0 ± 0.8
×	×	×	74.6 ± 0.7
-	-	-	74.7 ± 0.3

Table 4: Ablation study on three submodules with pretraining set $k = 3500$ and acquisition budget $B = 5000$ for SVHN. The last row corresponds to the random baseline.

Bilevel	Optimal Transport	RankNet	Accuracy
✓	✓	✓	$\mathbf{88.1 \pm 0.3}$
✓	✓	×	86.7 ± 0.2
✓	×	✓	87.8 ± 0.3
✓	×	×	86.5 ± 0.3
×	✓	✓	86.1 ± 0.2
×	✓	×	87.8 ± 0.2
×	×	✓	87.5 ± 0.1
×	×	×	86.1 ± 0.2
-	-	-	86.5 ± 0.3

of optimal transport distance, regardless of the scale of λ_{OT}, for various datasets of interest. Regardless of datasets and classification networks architecture, the incorporation of optimal transport distance finds utility in reducing generalization error, measured by the increase of validation set accuracy. Even though λ_{OT} can be a hard hyperparameter for fine-tuning,



(a) FashionMNIST (b) MNIST

(c) CIFAR10 (d) SVHN

Figure 5: Different choices of λ_{OT} for pretraining set size $k = 200$ for (a) and (b) and $k = 2500$ for (c) and (d) by different acquisition budget

Figure 5 suggests final validation set accuracy for $\lambda_{OT} \neq 0$ is higher than its counterpart for $\lambda_{OT} = 0$.

B.2.4 Runtime Analysis

All models are trained using NVIDIA A40 GPU with 48GB. To increase the running speed of our experiments, we use data parallelism on multiple GPUs in implementations. The time recorded below is for Pytorch training with 2 GPUs. As stated in main text, all the experiments are repeated for 10 trials to reduce the training stochasticity. We fix $k = 2500$ and $B = 5000$ for CIFAR10 and SVHN with $n = 30$ utility samples collected per batch with $\tau_1 = 2$, $b = 1000$ and $k_1 = 500$ for pretraining stage with each batch trained for 20 epochs. For CIFAR10, we collect 500 pairs of utility samples for $\hat{u}$ offline training with roughly 3 hours and 20 minutes. Then, the total training time for both pretraining and acquisition stage is 1 hour and 20 minutes with pretraining stage 50 minutes and acquisition stage 30 minutes. For SVHN, we collect 500 pairs of utility samples for offline training with roughly 1 hour and 40 minutes. Then, the total training time for both pretraining and acquisition stage is roughly 1 hour with pretraining stage 29 minutes and acquisition stage 34 minutes.

We fix $k = 200$ and $B = 500$ for MNIST and FashionMNIST with $n = 50$ utility samples collected per batch with $\tau_1 = 3$, $b = 50$ and $k_1 = 50$ for pretraining stage with each batch

trained for 20 epochs. For MNIST, we first randomly collect 500 pairs of utility samples and the total training time for utility model $\hat{u}$ for offline training is 50 minutes. Then, the total training time for both pretraining and acquisition stage is 59 minutes with pretraining stage 39 minutes and acquisition stage 20 minutes. For FashionMNIST, the offline training for utility model $\hat{u}$ is 50 minutes for 500 pairs of utility samples. The total training time for both pretraining and acquisition stage is 50 minutes with pretraining stage 32 minutes and acquisition stage 18 minutes.