Labeled TrustSet Guided: Batch Active Learning with Reinforcement Learning

Guofeng Cui, Yang Liu, Pichao Wang, Hankai Hsu, Xiaohang Sun, Xiang Hao, and Zhu Liu
Prime Video, Amazon, Seattle, USA
Email: {gfcui404, pichaowang}@gmail.com, {yangnliu, hankhsu, sunking, xianghao, zhuzliu}@amazon.com

***Abstract*—Batch active learning (BAL) is a crucial technique for reducing labeling costs and improving data efficiency in training large-scale deep learning models. Traditional BAL methods often rely on metrics like Mahalanobis Distance to balance uncertainty and diversity when selecting data for annotation. However, these methods predominantly focus on the distribution of unlabeled data and fail to leverage feedback from labeled data or the model's performance. To address these limitations, we introduce TrustSet, a novel approach that selects the most informative data from the labeled dataset, ensuring a balanced class distribution to mitigate the long-tail problem. Unlike CoreSet, which focuses on maintaining the overall data distribution, TrustSet optimizes the model's performance by pruning redundant data and using label information to refine the selection process. To extend the benefits of TrustSet to the unlabeled pool, we propose a reinforcement learning (RL)-based sampling policy that approximates the selection of high-quality TrustSet candidates from the unlabeled data. Combining TrustSet and RL, we introduce the Batch Reinforcement Active Learning with TrustSet (BRAL-T) framework. BRAL-T achieves state-of-the-art results across 10 image classification benchmarks and 2 active fine-tuning tasks, demonstrating its effectiveness and efficiency in various domains.**

***Index Terms*—active learning, reinforcement learning**

I. Introduction

In the era of deep learning, large-scale labeled datasets are indispensable for training models on complex tasks. Active learning (AL) provides an efficient approach to reduce the labeling costs by intelligently selecting critical subsets from unlabeled data for annotation [1–3]. Batch active learning (BAL) [4], a variant of AL, further improves this process by selecting data points in groups (batches), thereby reducing the overhead associated with model retraining and oracle interactions.

In most modern BAL methods, the selection strategy is typically based on two factors: uncertainty and diversity. Uncertainty-based methods focus on choosing the most ambiguous or difficult data, which is likely to improve the model, but this often results in selecting redundant data that doesn't sufficiently cover the data distribution [5]. On the other hand, diversity-based methods aim to ensure a representative subset by covering as many different types of data as possible, but they may neglect critical uncertain samples near the decision boundaries. For instance, CoreSet [6] selects subsets that reflect the overall data distribution, ensuring diversity by minimizing the distance between the selected subset and the full dataset. While methods like Cluster-Margin [4] combine diversity and uncertainty to improve data selection, they still have limitations, such as overlooking feedback from the labeled dataset, ignoring class distribution, and potentially inheriting the long-tail distribution problem.

To address these challenges, we propose TrustSet, a novel data selection approach that distinguishes itself from CoreSet by emphasizing the utilization of label information. TrustSet focuses not only on ensuring diversity but also on selecting data that is most beneficial for improving the model's performance and releasing class imbalance problem. TrustSet differs from CoreSet in two ways:

Objective: TrustSet is designed to optimize the model's performance, with an explicit focus on improving accuracy and tackling the long-tail distribution problem by selecting crucial data that has a high potential to be forgotten by the model [7]. In contrast, CoreSet focuses on representing the full data distribution without directly considering the impact on the model's learning process, which can lead to inheriting undesirable distributional imbalances.

Data Source: TrustSet leverages labeled data, utilizing ground truth labels to prune redundant and noisy data and ensure that the selected subset is balanced across classes. This approach contrasts with CoreSet, which selects data purely from the unlabeled pool, without considering feedback from the trained model. As a result, CoreSet-based methods can miss the opportunity to incorporate critical information about the model's current performance, potentially leading to suboptimal data selections.

TrustSet's balanced class distribution ensures better handling of the long-tail distribution problem, where underrepresented classes are more likely to be included in the training process. To construct TrustSet, we use the GradNd method [8], which ranks data based on the gradient norms of model updates, prioritizing data points that contribute most to model learning. Furthermore, to improve the data quality, we incorporate SuperLoss [9], which follows curriculum learning strategy to assign higher importance to easier data in early training stages, while still considering difficult samples later.

However, extending the TrustSet concept to the unlabeled data pool presents a challenge, as the selection process requires label information. To overcome this, we introduce an RL-

*Guofeng Cui and Pichao Wang are now with Nvidia. This work was done while they were at Amazon

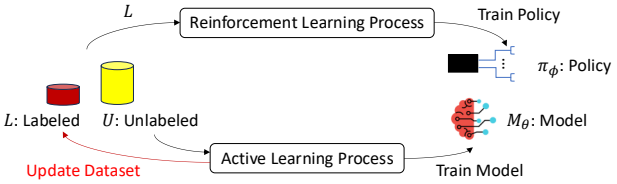

Fig. 1: An overview of the BRAL-T framework, which consists of two main processes: (1)**Active Learning**, where data is selected for labeling and used to train the model. (2)**Reinforcement Learning**, where a policy network learns to approximate TrustSet selection from unlabeled data pool.

based policy for approximating the selection of high-potential TrustSet candidates from the unlabeled data pool. Unlike previous RL-based active learning approaches, which often require frequent retraining of the model and rely heavily on complex reward structures [10, 11], our method minimizes retraining costs by leveraging TrustSet to guide the reinforcement learning process.

To this end, we propose a novel batch active learning framework called BRAL-T (**B**atch **R**einforcement **A**ctive **L**earning with **T**rustSet extraction), which integrates TrustSet and RL-based policies for efficient data selection. The framework consists of two primary components: (1) TrustSet extraction, which ensures that the labeled dataset contributes optimally to model performance and maintains a balanced class distribution, and (2) RL-based subset selection, where a learned policy selects from the unlabeled data pool to approximate TrustSet. This significantly reduces the need for repeated oracle queries and model retraining. As shown in Fig.1, BRAL-T is implemented with two processes: reinforcement learning (RL) for policy training and active learning (AL) for model training.

Our contributions are summarized as follows:

- We introduce TrustSet, a novel method for data selection that leverages label information to balance uncertainty, diversity, and class distribution, thus releasing class imbalanced issue and improving performance of AL.
- We develop an RL-based data selection policy that bridges the gap between TrustSet's label dependency and the unlabeled setting of active learning, allowing for more efficient and targeted data selection.
- We propose BRAL-T, a new batch active learning framework that integrates TrustSet and RL to reduce the computational burden of active learning while improving model performance. We demonstrate that BRAL-T achieves state-of-the-art performance across multiple image classification and active fine-tuning tasks.

II. Related Work

Batch Active Learning: Active learning aims to reduce labeling costs by selecting informative data based on uncertainty and diversity. Recent works [12–18] highlight the uncertainty-diversity trade-off, where uncertainty-focused methods risk redundancy, and diversity-based methods may overlook critical uncertain samples. Batch active learning approaches [4, 11, 16, 19, 20] concentrate on batch sampling to reduce costs and preserve subset distribution. They prefer batch data processing over individual sample handling. BatchBald [19] addressed the lack of joint informativeness in batch sampling with an entropy-based selection. Cluster-Margin [4] used Hierarchical Agglomerative Clustering to identify highly uncertain samples at scale. Despite their success in computer vision tasks, existing methods overlook feedback from selected samples, such as accuracy changes, which could be crucial for refining data sampling strategies.

Active Learning with RL: RL has been explored to learn data selection policies [10–12, 21–23]. Some approaches [10, 21–23] defined rewards based on target model metrics (e.g., accuracy, AUROC), requiring frequent retraining and suffering from credit assignment issues. Instead, [12] used the Mahalanobis distance for reward definition, while TAILOR [11] formulated class balance as a reward signal. However, the former is task-specific (Re-ID), and the latter does not directly align with model accuracy. To address these limitations, we propose an RL-based active learning approach that avoids retraining while maintaining high correlation with task performance.

III. Problem Definition

In this section, we formally define the active learning problem in batch setting, following [20], and introduce the TrustSet selection problem. We consider an $\mathbb{C}$-class classification task over a compact input space $\mathcal{X}$ and a label space $\mathcal{Y} = \{1, \ldots, \mathbb{C}\}$. Our goal is to train a target model M_θ with parameters θ to optimize a loss function $\ell(M_\theta(\mathbf{x}), y) : \mathcal{X} \times \mathcal{Y} \to R$, where $\hat{y} = M_\theta(\mathbf{x})$ refers to the predicted category by the target model and cross entropy loss is widely used as ℓ in classification task. In practice, we assume a large collection of data points sampled i.i.d. over the space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ as $\{\mathbf{x}_k, y_k\}_{k\in[n]} \sim p_\mathcal{Z}$, where n refers to the total amount of data points and $[n] = \{1, 2, \ldots, n\}$.

For active learning problem, we further define the labeled dataset with $|L|$ data points as $L = \{\mathbf{x}_k, y_k\}_k^{|L|}$ and the unlabeled data pool with $|U|$ data points as $U = \{\mathbf{x}_k\}_k^{|U|}$. In general, $|L| \ll |U|$ and $L \cap U = \emptyset$. For the i-th active learning iteration, the labeled and unlabeled dataset are defined as L_i and U_i respectively. We aim to select a data subset $S_{U_i} \subseteq U_i$ to be labeled by an oracle and added to the current labeled dataset L_i forming an enhanced labeled dataset $L_{i+1} = L_i \cup S_{U_i}$. We abbreviate S_{U_i} as S_i for readability in the rest of the paper. Training on L_{i+1}, the model $M_{\theta_{L_{i+1}}}$ with the trained parameter $\theta_{L_{i+1}}$ is expected to achieve the best performance across all possible choices of S_i which can be formulated as the following Eq.1:

$$S_i^* = \underset{S_i \subseteq U_i : |S_i| \leq b}{\arg\min} \; E_{\mathbf{x}, y \sim p_\mathcal{Z}} [\ell(M_{\theta_{L_{i+1}}}(\mathbf{x}), y)] \tag{1}$$

where S_i^* refers to the optimal data subset for active learning and b refers to the size of data subset that needs to be selected in each iteration. As a result, Eq.1 indicates that we aim to

select a data subset S_i from U_i to enhance L_i such that the trained model $M_{\theta_{L_{i+1}}}$ achieves the minimal loss value.

Directly solving the above optimization problem is challenging due to the large number of possible choice of S_i from U_i. To address this, we consider an ideal scenario where label information for U_i is available and analyze the entire dataset $D = L_i \cup U_i$ to identify the most important samples that contribute to model training. Formally, we define the TrustSet T_D to be the important data of D as the following Eq.2:

$$T_D = \underset{S_D \subseteq D: |S_D| \leq b_T}{\arg\min} E_{\mathbf{x},y\sim p_{\mathcal{Z}}}[\ell(M_{\theta_{S_D}}(\mathbf{x}), y)] \quad \text{s.t. balance}(S_D) \tag{2}$$

where S_D refers to a data subset selected from D and we require S_D to be balanced across $\mathbb{C}$ categories for TrustSet selection to alleviate long-tail distribution problem; $M_{\theta_{S_D}}$ refers to the model trained on S_D and b_T refers to the predefined size of TrustSet. Eq.2 indicates that given the limited size of data subset, T_D contains the most useful information from the dataset to train the model. Combining with Eq.1, we instead select $\tilde{S}_i$ from the unlabeled data pool U_i for active learning as:

$$\tilde{S}_i = \underset{S_i \subseteq U_i, |S_i| \leq b}{\arg\min} d(S_i, T_D \cap U_i) \tag{3}$$

where $d(\cdot,\cdot)$ indicates a statistical distance function (e.g. Wasserstein Distance) for two distributions and we aim to select a data subset that has a distribution similar to $T_D \cap U_i$. As T_D contains the most significant data points for model training, enhancing L_i with such data points could also benefit the performance of the trained model. As a result, $\tilde{S}_i$, which approximates $T_D \cap U_i$, also contributes to a significant improvement of model training. Compared to Eq.1, TrustSet is more reliable due to the use of annotation labels and the balance requirement releases the class imbalance problem of collected labeled dataset. Additionally, Eq.2 can be more straightforwardly solved using data pruning methods [8].

However, label information of U_i is not available under the active learning assumption. To solve this problem, we train a data selection policy π_{ϕ_i} with RL method on the labeled dataset L_i, learning to select $\tilde{S}_i$ as an approximation of $T_D \cap U_i$. In the i-th iteration, we create an environment similar to the active learning setting by randomly sampling two subsets $\mathcal{L}$ and $\mathcal{U}$ from labeled dataset L_i, where the labels of $\mathcal{L}$ are retained to simulate labeled dataset while the labels of $\mathcal{U}$ are omitted to simulate unlabeled dataset. The whole dataset in this environment is then defined as $\mathcal{D} = \mathcal{L} \cup \mathcal{U}$, ensuring $\mathcal{L} \cap \mathcal{U} = \emptyset$ and $|\mathcal{L}| \ll |\mathcal{U}|$. Although we omit the labels of $\mathcal{U}$ for active learning purpose, we can still leverage the label to extract $T_{\mathcal{D}}$. Taking $\mathcal{L}$ and $\mathcal{U}$ as input, π_{ϕ_i} is trained to select a subset $\mathcal{S}$ from $\mathcal{U}$, with the reward defined as:

$$R = -d(\mathcal{S}, T_{\mathcal{D}} \cap \mathcal{U}) \tag{4}$$

where $\mathcal{S} = \pi_{\phi_i}(\mathcal{L},\mathcal{U})$ and we optimize the parameters ϕ_i to minimize the statistical distance between $T_D \cap \mathcal{U}$ and $\mathcal{S}$. In this way, π_{ϕ_i} learns to select data from the unlabeled dataset that has a high potential to be included in the TrustSet based on the feature space. After training, π_{ϕ_i} is applied to the real unlabeled data pool U_i to select $S_i = \pi_{\phi_i}(L_i, U_i)$.

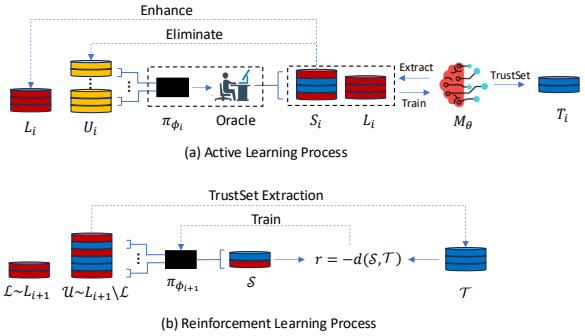

Fig. 2: Details of BRAL-T: active learning and RL processes.

In general, for each active learning iteration, we solve Eq.1 in two processes as shown in Fig.2. **In the active learning process**, data subset S_i is selected by $\pi_{\phi_i}(L_i, U_i)$ and passed to the oracle for annotation. The model M_θ is then trained on the enhanced labeled dataset L_{i+1}. Moreover, the new unlabeled data pool U_{i+1} is achieved by eliminating S_i from U_i and the TrustSet T_i is extracted from L_{i+1} by solving Eq.2 to facilitate the next RL process. **In the RL process**, we create the environment based on L_i and train π_{ϕ_i} using the reward function defined in Eq.4.

IV. Method

In this section, we introduce BRAL-T framework in detail. We first introduce a TrustSet construction method based on GradNd score [8]. Then, we illustrate the details of the RL module and describe how we use the learned policy to select subsets from the unlabeled pool.

A. TrustSet

In general, the TrustSet should retain important data and tend to be class-balanced. However, it is almost impossible to directly solve Eq.2 due to the large size of labeled dataset and time-consuming of $M_{\theta_{S_D}}$ training. As a result, we introduce a TrustSet extraction method based on the GradNd score [8] by analyzing the performance of model trained on entire trainset L rather than selected subset S. This score is defined as the expected value of the gradient norm term with respect to a differentiable model and a data sample x:

$$GradNd = E\|\sum_{k=1}^{K} \nabla_{M_\theta^{(k)}} \ell(M_\theta(x), y)^T \nabla_\theta M_\theta^{(k)}(x)\| \tag{5}$$

In this equation, K denotes the number of logits, $M_\theta(x) \in \mathbb{R}^K$ refers to the output of model and $M_\theta^{(k)}(x)$ represents the result of the k-th logit from the model M_θ. For instance, in an image classification task, ℓ represents the cross-entropy loss, $K = \mathbb{C}$ is the number of categories, and $M_\theta^{(k)}(x)$ is the logit output for the k-th category. Data samples that result in a large gradient value tend to contain information that the model has not yet learned, as the model would update significantly based on such data. As demonstrated by the experimental analysis from [8],

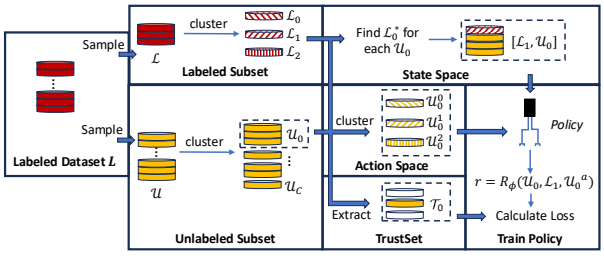

Fig. 3: Reinforcement learning process. We use $\mathcal{U}_0$ as an example for process in the figure.

data with a higher GradNd score tend to be forgotten samples for the target model during training and are more important for further training. However, the GradNd score might lead to a class imbalance problem when the data subset primarily contains difficult images for certain categories. To mitigate the long-tail distribution problem, we sort data by class using the GradNd score and select the top-N data for each category. For the image classification task, we follow [8] in omitting the term $\nabla\theta M_\theta^{(k)}(x)$ from Eq 5 and calculate the EL2N score to approximate GradNd.

Curriculum Learning: Data with high GradNd scores tend to be difficult and uncertain samples. As suggested by previous works [4, 16, 18], a training set focusing on uncertainty could result in high redundancy and fail to train a model that captures general features. We reconsider this issue from another important perspective. Difficult samples contain noise that can interfere with model predictions and increase the difficulty for the model to learn the boundaries between categories. With a limited amount of data, easy examples could help the model capture features and cluster data within the same category. Following the principles of curriculum learning [9, 24], we assign larger weights to easier data samples in the early active learning iterations and leverage Super Loss [9] on top of the task loss ℓ. For each data sample (x, y), the super loss ℓ_s is defined as:

$$\ell_s(M_\theta(x), y) = (\ell(M_\theta(x), y) - \tau)\sigma + \lambda(\log\sigma)^2 \quad (6)$$

where τ is the threshold for separating easy and hard samples, and λ is the weight of the regularization term. Both τ and λ are hyperparameters, while σ is learnable and indicates the weight assigned to the task loss. To minimize ℓ_s, data with task loss $\ell < \tau$ will be assigned a larger weight σ, and data with $\ell > \tau$ will be assigned a smaller weight σ. Since model training on uncertain data typically results in larger losses compared to easier data, super loss adaptively adjusts the weight for data samples. Meanwhile, the scale of σ is determined by λ. As λ increases, the value of σ tends to be 1 and has less effect on the task loss. Specifically, when $\lambda \to \infty$, σ will always be 1 to minimize the regularization term, making ℓ_s equal to $\ell - \tau$. In the following sections, T_D refers to the TrustSet with Super Loss unless explicitly stated.

B. Reinforcement Learning

The TrustSet is collected with label information to ensure class balance and improved reliability. However, during the active learning process, the policy needs to be applied to select a subset from the unlabeled data pool U_i. To address this, we create an environment with similar conditions and apply reinforcement learning to train a policy for subset selection, where the TrustSet serves as the target subset. In the remainder of this section, we first define the state, action, and reward for the reinforcement learning task in general, followed by an illustration of the overall process.

State $(\mathcal{L}_c^*, \mathcal{U}_c)$**.** We randomly sample $\mathcal{L}$ as a labeled dataset and $\mathcal{U}$ as an unlabeled data pool from L_i to train the policy π_{ϕ_i}. For each sampled dataset $\mathcal{D} = \mathcal{L} \cup \mathcal{U}$, we extract $T_\mathcal{D}$ and calculate $T_\mathcal{D} \cap \mathcal{U}$ as the target selected subset. For convenience, we define $\mathcal{T} = T_\mathcal{D} \cap \mathcal{U}$. Using all data in $\mathcal{L}$ and $\mathcal{U}$ as input is computationally expensive and challenging for learning an effective policy. It is beneficial to have alternative representations. Since in classification task, data tend to cluster based on predicted categories in feature space and $\mathcal{T}$ is distributed across all clusters, it is more reasonable to predict $\mathcal{T}$ using clusters as states. Thus, we define the state space as $(\mathcal{L}_c^*, \mathcal{U}_c)$, where $\mathcal{U}_c$ refers to the c-th cluster from the unlabeled data pool $\mathcal{U}$, and $\mathcal{L}_c^*$ is a cluster from the labeled dataset $\mathcal{L}$ defined as:

$$\mathcal{L}_c^* = \arg\min_m d(\mathcal{L}_m, \mathcal{U}_c) \quad (7)$$

In this equation, $\mathcal{L}_m$ refers to the m-th cluster from $\mathcal{L}$, and $\mathcal{L}_c^*$ is the closest labeled cluster to $\mathcal{U}_c$ based on the distance function $d(\cdot,\cdot)$. We use the Wasserstein Distance in our reinforcement learning process. For each $(\mathcal{L},\mathcal{U})$ sample, there are C states, where C is the number of clusters in $\mathcal{U}$. For improved efficiency, we extract stochastic features of clusters as input for the policy, specifically using mean and variance as $[E[\mathcal{L}_c^*], Var[\mathcal{L}_c^*], E[\mathcal{U}_c], Var[\mathcal{U}_c]]$.

Action $\mathcal{U}_c^a$**.** As data in the TrustSet tend to group together by cluster, we further divide $\mathcal{U}_c$ into A_c data groups, denoted as $\{\mathcal{U}_c^a\}_{a=1}^{A_c}$. Given $(\mathcal{L}_c^*, \mathcal{U}_c)$ as input, the policy selects the top clusters within $\{\mathcal{U}_c^a\}_{a=1}^{A_c}$ that have high potential to be included in the TrustSet. Consequently, $\{\mathcal{U}_c^a\}_{a=1}^{A_c}$ represents the candidate action space for each state, and the union of the selected actions forms the final selected data subset $\mathcal{S}$.

Reward R**.** Since different $\mathcal{U}_c^a$ contain varying numbers of data points, for a fixed size of $\mathcal{S}$, we need to select a varying number of $\mathcal{U}_c^a$. It is more general to define the reward based on $\mathcal{U}_c^a$ rather than $\mathcal{S}$. We set the reward as the negative Wasserstein distance between $\mathcal{U}_c^a$ and the sub-TrustSet $\mathcal{T}_c = \mathcal{T} \cap \mathcal{U}_c$ as:

$$R = -d(\mathcal{U}_c^a, \mathcal{T}_c) \quad (8)$$

where $\mathcal{U}_c^a$ closer in distribution to $\mathcal{T}_c$ receives better reward.

We follow the DQN method [25] and illustrate the overall RL process in Fig.3. Given the dataset $\mathcal{L}$ and $\mathcal{U}$, we pass them through the target model M_{θ_i} to obtain the feature space. To construct the input state for the policy, we cluster the features of $\mathcal{L}$ into M clusters, denoted as $\{\mathcal{L}_m\}_{m=1}^M$ and the features of

$\mathcal{U}$ into C clusters as $\{\mathcal{U}_c\}_{c=1}^{C}$. To generate candidate actions, we further divide $\mathcal{U}_c$ into A_c clusters, denoted as $\{\mathcal{U}_c^a\}_{a=1}^{A_c}$. Meanwhile, we extract the TrustSet for each cluster as $\mathcal{T}_c = \mathcal{T} \cap \mathcal{U}_c$.

Unlike traditional RL tasks, we consider the future effect in curriculum learning and TrustSet selection, focusing only on the next timestep for policy training. As a result, training the Q-function is equivalent to training the reward function R_ϕ as:

$$\tilde{R} = R_\phi(\mathcal{U}_c, \mathcal{L}_c^*, \mathcal{U}_c^a) \tag{9}$$

where $\tilde{R}$ refers to the predicted reward value by R_ϕ and ϕ refers to the parameter of the reward function. And ϕ is updated by the Mean Square Error (MSE) loss as:

$$l_{RL} = E_{(\mathcal{U}_c, \mathcal{U}_c^a)} \|R - \tilde{R}\| \tag{10}$$

During reinforcement learning, we calculate the reward for all candidate actions and states to optimize the reward function R_ϕ. During the active learning process, for each unlabeled cluster U_c in the real unlabeled data pool U, we predict the reward for all candidate actions U_c^a and select them in descending order based on the reward score until the fixed size of subset selection is satisfied.

It is worthwhile to note that for each $(\mathcal{L},\mathcal{U})$ sampled from L, M_θ needs to be retrained on $\mathcal{D} = \mathcal{L} \cup \mathcal{U}$ to extract the TrustSet $T_\mathcal{D}$ as well as $\mathcal{T}$, since the most important data for the model may vary depending on the labeled training set. To avoid the time-consuming process of frequent retraining, we approximate the extraction of $T_\mathcal{D}$ by reusing $M_{\theta_{L_i}}$ which has been trained on L_i in the i-th active learning iteration. This is justified by the fact that $\mathcal{D}$ is randomly sampled from L_i and our main objective is to enhance L_i based on the performance of $M_{\theta_{L_i}}$ in the i-th active learning iteration. The only requirement is to retrain the policy from scratch for each active learning iteration. In practice, we extract T_i from the entire labeled set L_i and extract $\mathcal{T}$ by taking the intersection between $\mathcal{U}$ and T_i as $\mathcal{T} = T_i \bigcap \mathcal{U}$ for efficiency.

V. Experiment

In this section, we evaluate our proposed BRAL-T method on the image classification task and compare our results with previous active learning baselines, following the experimental settings of [1]. Additionally, we also evaluate BRAL-T on the active fine-tuning task [26] and compare it with the current state-of-the-art method, ActiveFT.

A. Image Classification Results

Datasets: We evaluated BRAL-T on the image classification task across 8 benchmarks, including Cifar10, Cifar100 [27], Cifar10-imb, EMNIST [28], FashionMNIST [29], BreakHis [30], Pneumonia-MNIST [31] and Waterbird [32, 33]. To create the Cifar10-imb dataset, we subsampled the training set of Cifar10 with ratios of 1:2:...:10 for classes 0 through 9.

Baselines: We compared BRAL-T with three baselines, LossPrediction [34], WAAL [35] and RandomSample. LossPrediction employs an additional module that predicts the loss for each data point. WAAL adopts min-max loss to better distinguish labeled and unlabeled samples while searching unlabeled batch with higher diversity than labeled samples. According to the experiments in [1], among all the methods, LossPrediction and WAAL achieve best results in 6 benchmarks and competitive results in other 2 benchmarks, therefore we select them as our baselines. For RandomSample, we randomly selected a subset from the unlabeled dataset in each active learning iteration. Besides, we visualized accuracy-budget curve on Cifar10, Cifar10-imb, Cifar100 and FashionMNIST benchmarks and compared with LossPrediction, WAAL, VAAL [14], BADGE [13], CoreSet [1], ClusterMargin [4], BALD [36] and KMeans [13]. To ensure a fair comparison, we used ResNet18 [37] as the target model.

Evaluation Metrics: For all benchmarks, we report evaluation results using two metrics: *area under the budget curve* (AUBC) [38, 39] and *final accuracy* (F-acc). AUBC refers to the area under the accuracy-budget curve. Methods with a higher AUBC score achieve better overall performance across different sizes of the training set. F-acc refers to the final accuracy achieved after the budget Q is exhausted. The experiments for BRAL-T and the baselines were repeated for 3 trials under different random seeds, and the average of the evaluation results are reported.

Experiment Results: The experimental results on 8 benchmarks are presented in Table I. BRAL-T significantly outperforms RandomSample under all benchmarks. Compared with WAAL and LossPrediction, BRAL-T achieves better AUBC as well as F-acc on all benchmarks. In Fig.4, we visualize the accuracy-budget curves of BRAL-T and baselines on 4 benchmarks. BRAL-T consistently achieves higher accuracy throughout the entire active learning process for Cifar100 and FashionMNIST. In early active learning iterations of Cifar10 and Cifar10-imb, BRAL-T has a bit worse accuracy compared with WAAL, the reasons of which could be attributed to WAAL's emphasis on diversity. However, without adequate consideration for uncertainty cause performance diminishing of WAAL when the size of labeled dataset increases. As comparison, LossPrediction focuses solely on uncertain data with high predicted loss, neglecting the diversity of the selected subset, which results in bad performance in early stage.

B. Active Learning on More Long-tail Datasets

Dataset. Besides the aforementioned benchmarks, in this section, we focus on long-tail datasets, including CIFAR10-LT and CIFAR100-LT. Both datasets are subsampled from CIFAR datasets and the number of samples within each classes decreases exponentially with factor within 10 and 100. Specifically, we consider 10, 20 and 50 in our experiments. The test images of CIFAR10-LT and CIFAR100-LT are the same as those in CIFAR10 and CIFAR100 datasets respectively. Both the two benchmarks are open-source and can be accessed through huggingface *tomas-gajarsky/cifar10-lt* and *tomas-gajarsky/cifar10-lt*.

Experiment Results. Besides LossPrediction, WAAL and RandomSample, we also compare BRAL-T with SIMI-

Methods	FashionMNIST		EMNIST		CIFAR10		CIFAR100	
	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc
LossPrediction	0.859	0.888	0.762	0.793	0.837	0.911	0.481	0.655
WAAL	0.861	0.891	0.808	0.831	0.842	0.883	0.460	0.594
RandomSample	0.844	0.874	0.804	0.828	0.832	0.902	0.517	0.650
BRAL-T	**0.863**	**0.894**	**0.813**	**0.833**	**0.847**	**0.916**	**0.525**	**0.662**
Benchmarks	**Cifar10-imb**		**BreakHis**		**Pneum.MNIST**		**Waterbird**	
	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc
LossPrediction	0.748	0.848	0.834	0.844	0.732	0.870	0.588	0.586
WAAL	0.752	0.799	0.836	0.855	0.640	0.870	0.525	0.506
RandomSample	0.710	0.810	0.834	0.832	0.706	0.652	0.586	0.502
BRAL-T	**0.762**	**0.851**	**0.849**	**0.868**	**0.738**	**0.883**	**0.606**	**0.618**

TABLE I: Experiment results of image classification task on 8 benchmarks

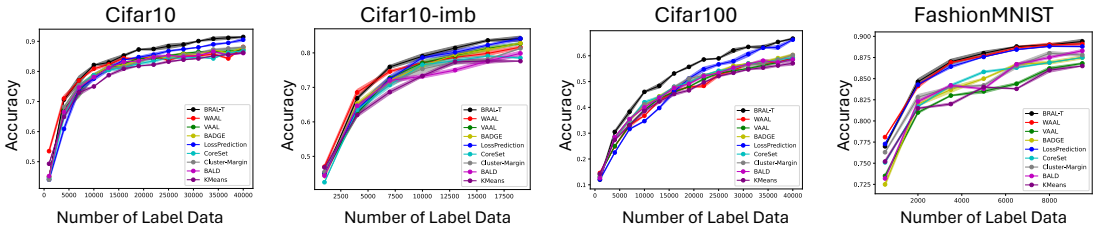

Fig. 4: Visualization of experiment results on Cifar10, Cifar10-imb, Cifar100 and FashionMNIST.

Methods	Cifar10-LT-r10		Cifar10-LT-r20		Cifar10-LT-r50		Cifar100-LT-r10		Cifar100-LT-r20		Cifar100-LT-r50	
	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc	AUBC	F-acc
TiDAL	0.510	0.619	0.435	0.583	**0.343**	0.460	0.398	0.450	0.365	0.420	0.320	0.373
SIMILAR	0.507	0.654	0.431	0.566	0.340	0.449	0.405	0.514	0.366	0.460	0.319	0.373
LossPrediction	0.478	0.679	0.413	0.585	0.322	0.494	0.424	0.516	0.380	0.460	0.323	0.390
WAAL	0.510	0.623	0.435	0.589	0.340	**0.497**	0.381	0.458	0.342	0.412	0.302	0.345
RandomSample	0.476	0.686	0.397	0.587	0.303	0.484	0.382	0.463	0.343	0.410	0.302	0.351
BRAL-T	**0.512**	**0.686**	**0.440**	**0.614**	0.322	0.490	**0.432**	**0.517**	**0.384**	**0.462**	**0.328**	**0.391**

TABLE II: Experiment results on Cifar10-LT and Cifar100-LT datasets.

LAR [40] and TiDAL [41] which are designed for active learning in imbalanced dataset. As shown in Table II, except for CIFAR10-LT-r50 benchmark, BRAL-T achieves the best AUBC as well as F-acc results. As pseudo label is not reliable especially when target model might be overconfident in long-tail dataset, BRAL-T selects informative data with ground-truth label to construct TrustSet which is more reliable to reflect whether target model has sufficiently learnt from related samples. As a result, BRAL-T always performs better than SIMILAR. Moreover, compared with LossPrediction and WAAL, we encourage TrustSet to be balanced which releases the category bias problem in long-tail distribution and contributes to the success of BRAL-T.

C. Active Learning for Finetuning Results

Experiment Setting: We adhere to the settings of [26] and use Deit-Small [42], pretrained with the DINO [43] framework on ImageNet-1k, as the target model. We chose two datasets for fine-tuning: Cifar10-imb and TinyImageNet [44], resizing all images to 224×224. For more implementation details, we utilize ActiveFT [26] to select an 1% subset as the initial labeled dataset and select an additional 1% of data for each active learning iteration. The pretrained model is fine-tuned using the SGD optimizer for 1000 epochs with a batch size of 512. Cosine learning rate decay is applied during the fine-tuning phase of each active learning iteration.

Methods	Cifar10-imb		TinyImageNet	
	2%	3%	2%	3%
RandomSample	0.841	0.856	0.213	0.348
ActiveFT	0.838	0.851	0.289	0.359
BRAL-T	**0.852**	**0.865**	**0.300**	**0.392**

TABLE III: Experiment results of Active Finetuning task.

Baseline	Cifar10		Cifar100	
	AUBC	F-acc	AUBC	F-acc
PseudoScore	0.842	0.908	0.486	0.661
BRAL-DiffSet	0.843	0.909	0.521	0.652
BRAL-T w/o CL	0.845	0.906	0.522	0.662
RandomSample	0.832	0.902	0.517	0.650
BRAL-T	**0.847**	**0.916**	**0.525**	**0.662**

TABLE IV: Ablation study result on Cifar10 and Cifar100.

Experiment Results: Experiments for all methods were repeated across 3 trials, and the average results are reported in Table III. BRAL-T significantly outperforms the other baselines. ActiveFT aims to select a data subset with a

distribution similar to that of the unlabeled data pool. However, if the unlabeled data pool suffers from problems like long-tail distribution or contains a large number of noisy data samples, the selected subset will likely encounter the same issues. For instance, Cifar10-imb is a class-imbalanced dataset, and TinyImageNet includes images that are difficult to classify. In contrast, TrustSets are defined to be class-balanced, and curriculum learning encourages the selection of easy or less-noisy data samples in TrustSets. Being trained with the distribution of TrustSets, the policy with the reward function R_ϕ learns to select data samples that provide greater benefits for model training.

D. Ablation Study

Experiment Setting: To further evaluate BRAL-T, we conducted ablation studies to demonstrate the benefits of the proposed modules by considering three baselines. For **PseudoScore**, instead of training an RL policy, we assign pseudo-labels to the unlabeled data pool based on the category with the highest logit score. Data subset with top EL2N score will be selected during active learning. For **BRAL-DiffSet**, to show the effectiveness of TrustSet, We select the second-best data group instead of the best as TrustSet. For **BRAL-T w/o CL**, we remove curriculum learning and directly use the cross-entropy loss function to calculate the EL2N score.

Experiment Results: Table IV displays the results for the Cifar10 and Cifar100 datasets. BRAL-T surpasses PseudoScore in both AUBC and F-acc as pseudo-labels are often inaccurate, especially when the training set lacks sufficient data to train a high-performance classifier. In contrast, selecting the TrustSet based on the labeled dataset is more reliable. Compared to BRAL-DiffSet, BRAL-T also achieves better AUBC and F-acc scores. Base on the result, we can empirically prove the correlation between EL2N score and model accuracy. Moreover, the results demonstrate that the RL policy successfully learns to select potentially important data. Curriculum learning also plays a crucial role in the success of BRAL-T, which aids in selecting easy examples and enhances the performance of the target model in the initial stages, the removal of which leads to worse results in BRAL-T w/o CL.

VI. Conclusion

In summary, our RL-based Active Learning framework, BRAL-T, leverages TrustSet to more accurately evaluate distribution of labeled datasets and employs an RL policy to learn from TrustSet. BRAL-T benchmarked against 8 baselines across 8 image classification tasks, shows superior AUBC and F-acc performance. Moreover, in CIFAR-LT benchmarks, BRAL-T outperforms baselines to handle long-tail dataset. Additionally, its application in active fine-tuning tasks reveals new state-of-the-art results.

References

[1] X. Zhan, Q. Wang, K.-h. Huang, H. Xiong, D. Dou, and A. B. Chan, "A comparative survey of deep active learning," *arXiv preprint arXiv:2203.13450*, 2022.

[2] G. Németh and T. Matuszka, "Compute-efficient active learning," *arXiv preprint arXiv:2401.07639*, 2024.

[3] B. Safaei, V. Vibashan, C. M. de Melo, and V. M. Patel, "Entropic open-set active learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 5, 2024, pp. 4686–4694.

[4] 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*, vol. 34, pp. 11 933–11 944, 2021.

[5] 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.

[6] J. M. Phillips, "Coresets and sketches," in *Handbook of discrete and computational geometry*. Chapman and Hall/CRC, 2017, pp. 1269–1288.

[7] M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," *arXiv preprint arXiv:1812.05159*, 2018.

[8] M. Paul, S. Ganguli, and G. K. Dziugaite, "Deep learning on a data diet: Finding important examples early in training," *Advances in Neural Information Processing Systems*, vol. 34, pp. 20 596–20 607, 2021.

[9] T. Castells, P. Weinzaepfel, and J. Revaud, "Superloss: A generic loss for robust curriculum learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4308–4319, 2020.

[10] M. Fang, Y. Li, and T. Cohn, "Learning how to active learn: A deep reinforcement learning approach," *arXiv preprint arXiv:1708.02383*, 2017.

[11] J. Zhang, S. Shao, S. Verma, and R. Nowak, "Algorithm selection for deep active learning with imbalanced datasets," *arXiv preprint arXiv:2302.07317*, 2023.

[12] Z. Liu, J. Wang, S. Gong, H. Lu, and D. Tao, "Deep reinforcement active learning for human-in-the-loop person re-identification," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6122–6131.

[13] 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.

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

[15] K. Margatina, G. Vernikos, L. Barrault, and N. Aletras, "Active learning by acquiring contrastive examples," *arXiv preprint arXiv:2109.03764*, 2021.

[16] J. Ash, S. Goel, A. Krishnamurthy, and S. Kakade, "Gone fishing: Neural active learning with fisher embeddings," *Advances in Neural Information Processing Systems*, vol. 34, pp. 8927–8939, 2021.

[17] K. Kim, D. Park, K. I. Kim, and S. Y. Chun, "Task-aware variational adversarial active learning," in *Proceedings*

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 8166–8175.

[18] C. Gentile, Z. Wang, and T. Zhang, "Achieving minimax rates in pool-based batch active learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 7339–7367.

[19] A. Kirsch, J. Van Amersfoort, and Y. Gal, "Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning," *Advances in neural information processing systems*, vol. 32, 2019.

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

[21] J. Gong, Z. Fan, Q. Ke, H. Rahmani, and J. Liu, "Meta agent teaming active learning for pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 079–11 089.

[22] A. Smit, D. Vrabac, Y. He, A. Y. Ng, A. L. Beam, and P. Rajpurkar, "Medselect: Selective labeling for medical image classification combining meta-learning with deep reinforcement learning," *arXiv preprint arXiv:2103.14339*, 2021.

[23] A. Casanova, P. O. Pinheiro, N. Rostamzadeh, and C. J. Pal, "Reinforced active learning for image segmentation," *arXiv preprint arXiv:2002.06583*, 2020.

[24] Y.-P. Tang and S.-J. Huang, "Self-paced active learning: Query the right thing at the right time," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 5117–5124.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[26] Y. Xie, H. Lu, J. Yan, X. Yang, M. Tomizuka, and W. Zhan, "Active finetuning: Exploiting annotation budget in the pretraining-finetuning paradigm," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 715–23 724.

[27] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[28] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 2921–2926.

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

[30] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte, "A dataset for breast cancer histopathological image classification," *IEEE transactions on biomedical engineering*, vol. 63, no. 7, pp. 1455–1462, 2015.

[31] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, A. McKeown, G. Yang, X. Wu, F. Yan *et al.*, "Identifying medical diagnoses and treatable diseases by image-based deep learning," *cell*, vol. 172, no. 5, pp. 1122–1131, 2018.

[32] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang, "Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization," *arXiv preprint arXiv:1911.08731*, 2019.

[33] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao *et al.*, "Wilds: A benchmark of in-the-wild distribution shifts," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5637–5664.

[34] D. Yoo and I. S. Kweon, "Learning loss for active learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 93–102.

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

[36] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International conference on machine learning*. PMLR, 2017, pp. 1183–1192.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*. Springer, 2016, pp. 630–645.

[38] X. Zhan, Q. Li, and A. B. Chan, "Multiple-criteria based active learning with fixed-size determinantal point processes," *arXiv preprint arXiv:2107.01622*, 2021.

[39] X. Zhan, H. Liu, Q. Li, and A. B. Chan, "A comparative survey: Benchmarking for pool-based active learning." in *IJCAI*, 2021, pp. 4679–4686.

[40] S. Kothawade, N. Beck, K. Killamsetty, and R. Iyer, "Similar: Submodular information measures based active learning in realistic scenarios," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 685–18 697, 2021.

[41] S. M. Kye, K. Choi, H. Byun, and B. Chang, "Tidal: Learning training dynamics for active learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 22 335–22 345.

[42] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*. PMLR, 2021, pp. 10 347–10 357.

[43] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[44] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.