Towards Comparable Active Learning

April 14, 2023

1 Introduction

1.1 Contribution

2 Related Work

Version: Braindump

Many different algorithms have been proposed for active learning. In this work we focus on those approaches that have shown consistent results over the years as well as newer approaches that have demonstrated significant lifts in their initial experiments. AL algorithms can be categorized into two classes: Geometric approaches and uncertainty-based approaches. Geometric approaches include CoreSet [8] and TypiClust [3], which use clustering techniques to partition the data and then sample their unlabeled points based on the clusters. Uncertainty-based approaches include classic uncertainty sampling (based on Shannon-Entropy and the margin-score), BALD [4] and BADGE [1], which use metrics to measure the classifiers state.

Some previous work also aimed to provide a benchmark suite for active learning: The authors of [2] and [6] both focus on active learning in the image domain. While [2] discuss a new metric to measure AL performance, which they call "Label Efficiency" and provide experiments on many common configurations for data preparation, model training and other hyperparameters, [6] focuses on combined approaches of AL and semi-supervised learning to aid model training.

3 Overview

3.1 Problem Description / Delineation

Basic Classification

We assume a dataset $\mathcal{D} := (x_i, y_i)\,;\; i := 1 \dots N$ consisting of instances $x_i \in \mathbb{R}^M$ and corresponding $y_i \in \mathbb{R}^C$. For evaluation purposes we assume a held-out test set $\mathcal{D}^{\text{test}}$ with the same characteristics. We consider classification problems with one-hot encoded classes, hence C models the number of classes. To perform classification, a model $\hat{y}_\theta : \mathbb{R}^M \to \mathbb{R}^C$ is used. To fit the model, it is parameterized by θ and subjected to loss $\ell : \mathbb{R}^C \times \mathbb{R}^C \to \mathbb{R}$. For this work, categorical cross-entropy (CE) is used. For evaluating classification performance, we use accuracy on the test set $\text{Acc}(\mathcal{D}^{\text{test}}, \hat{y}_\theta)$.

Pool-based AL with single instances (non-batch setting)

To construct the active learning setting, we suppress the labels y_i of $\mathcal{D}$ to form the unlabeled pool $\mathcal{U} :=$

Paper	# Datasets	Domains	Scenarios	Oracle	RL Alg.
Beck et al. [2]	5	2	2	✓	-
Li et al. [6]	5	2	2	✓	-
Ours	5	2	2	✓	-

u_i ; $i := 1 \dots N$ and form and initial labeled pool $\mathcal{L}$ by uniformly sampling k number of instances per class from $\mathcal{U}$ and recovering their label. The result of this so-called "seeding" process is $\mathcal{L} := (u_i, y_i)$; $i := 1 \dots k * C$.
Active learning is defined as sequentially removing single instances $u^{(i)} \in \mathcal{U}^{(i)}$; $\mathcal{U}^{(i+1)} := \mathcal{U}^{(i)} \setminus \{u^{(i)}\}$, recovering their label $y^{(i)}$ and adding them to the labeled pool $\mathcal{L}^{(i+1)} := \mathcal{L}^{(i)} \cup (u^{(i)}, y^{(i)})$ until a fixed budget B is exhausted $i := 1 \dots B$. After each added instance the classification model is retrained according to section 4.2 and its performance is measured on the held-out test set $\mathcal{D}^{\text{test}}$. The quality of an active learning algorithm is evaluated by an "anytime" protocol that incorporates classification performance at every iteration, not just the final performance after the budget is exhausted. We employ the normalized area under the accuracy curve (AUC):

$$\text{auc}(\mathcal{U}, \mathcal{L}, \hat{y}, B) := \frac{1}{B} \sum_{i=1}^{B} \text{Acc}(y_{test}, \hat{y}_i(x_{test})) \tag{1}$$

Where $\hat{y}_i$ is the retrained classification model after the i-*th* instance was selected.

Framing AL as RL
We define the active learning process as an adapted reinforcement learning loop $(S, A, \tau, \Omega, \omega)$ where an environment iteratively will expose a state $s \in S$ to an agent Ω, which will choose actions $a \in A$. For each iteration i the environment samples a subset of size τ of unlabeled instances $u^{(i)} \underset{\tau}{\sim} \mathcal{U}^{(i)}$, constructs the state $s^{(i)} := \omega(u^{(i)})$ and presents it to the agent to select an action $a^{(i)} := \Omega(s^{(i)})$. The action $a^{(i)}$ is the index of the selected instance in $u^{(i)}$ out of all possible indices $A := [1 \dots \tau]$. This process is repeated B times $i := [1 \dots B]$.

Algorithm 1 Active Learning OLD

Require: $\mathcal{U}$ ▷ Unlabeled Pool
Require: τ ▷ Unlabeled Sample Size
Require: Ω ▷ AL Agent
Require: ω ▷ Environment State function
$\mathcal{L}^{(1)} \leftarrow \text{seed}(\mathcal{U})$ ▷ Create the initial labeled set
$\mathcal{U}^{(1)} \leftarrow \mathcal{U}$
for $i := 1 \dots B$ **do**
  $\text{acc}^{(i)} \leftarrow \text{Retrain}(\mathcal{L}^{(i)})$ ▷ $\text{Retrain}(\mathcal{L}^{(i)})$ is shorthand for $\text{Retrain}(\mathcal{L}^{(i)}, \mathcal{L}^{\text{test}}, \hat{y}_\theta, e^{\max})$
  $u^{(i)} \underset{\tau}{\sim} \mathcal{U}^{(i)}$
  $s^{(i)} \leftarrow \omega(u^{(i)})$
  $a^{(i)} \leftarrow \Omega(s^{(i)})$ ▷ $a^{(i)}$ is an index inside of $u^{(i)}$
  $y^{(i)} \leftarrow \text{label}(u_a^{(i)})$ ▷ $u_a^{(i)}$ is shorthand for $u_{a^{(i)}}^{(i)}$
  $\mathcal{L}^{(i+1)} \leftarrow \mathcal{L}^{(i)} \cup \{(u_a^{(i)}, y^{(i)})\}$
  $\mathcal{U}^{(i+1)} \leftarrow \mathcal{U}^{(i)} \setminus \{u_a^{(i)}\}$
end for
return $\frac{1}{B} \sum_{i=1}^{B} \text{acc}^{(i)}$

Algorithm 2 Retrain

Require: $\mathcal{L}$ ▷ Labeled Pool
Require: $\mathcal{L}^{\text{test}}$ ▷ Labeled Test Data
Require: $\hat{y}_\theta$ ▷ Classification Model
Require: $e^{\max}$ ▷ Maximum Epochs
$\text{loss}^* \leftarrow \infty$
for $i := 1 \dots e^{\max}$ **do**
  $\theta_{i+1} \leftarrow \theta_i - \eta \nabla_\theta \ell(\mathcal{L}, \hat{y}_\theta)$
  $\text{loss}_i \leftarrow \ell(\mathcal{L}^{\text{test}}, \hat{y}_\theta)$
  if $\text{loss}_i < \text{loss}^*$ **then**
    $\text{loss}^* \leftarrow \text{loss}_i$
  else
    Break
  end if
end for
return $\text{Acc}(\mathcal{L}^{\text{test}}, \hat{y}_\theta)$

3.2 Datasets

Version: 1.0
In this work we use 4 different datasets from two domains. For vector data, we use DNA and Splice (Source: [7]) and for image data we use FashionMNIST [9] and Cifar10 [5].
We would like to point out that these datasets can be considered "toy-datasets" and therefore not relevant for practical purposes. This might be true if we aimed to develop novel classification models on these dataset,

Algorithm 3 Oracle OLD

Require: $\mathcal{U}$ ▷ Unlabeled Pool
Require: τ ▷ Unlabeled Sample Size
Require: Ω ▷ AL Agent
Require: ω ▷ Environment State function
$\mathcal{L}^{(1)} \leftarrow \text{seed}(\mathcal{U})$ ▷ Create the initial labeled set
$\mathcal{U}^{(1)} \leftarrow \mathcal{U}$
for $i := 1 \dots B$ **do**
  $\text{acc}^{(i)} \leftarrow \text{Retrain}(\mathcal{L}^{(i)})$ ▷ $\text{Retrain}(\mathcal{L}^{(i)})$ is shorthand for $\text{Retrain}(\mathcal{L}^{(i)}, \mathcal{L}^{\text{test}}, \hat{y}_\theta, e^{\max})$
  $u^{(i)} \underset{\tau}{\sim} \mathcal{U}^{(i)}$
  $r* \leftarrow -\infty$
  $j* \leftarrow -1$
  for $j := 1 \dots \tau$ **do** ▷ Testing every unlabeled point
    $y^{(j)} \leftarrow \text{label}(u_j^{(i)})$
    $\mathcal{L}^{(j)} \leftarrow \mathcal{L}^{(i)} \cup \{(u_j^{(i)}, y^{(j)})\}$
    $\text{acc}^{(j)} \leftarrow \text{Retrain}(\mathcal{L}^{(j)})$
    $r^{(j)} \leftarrow \text{acc}^{(j)} - \text{acc}^{(i)}$
    if $r^{(j)} > r^*$ **then** ▷ Select point with largest increase in performance
      $r* \leftarrow r^{(j)}$
      $j* \leftarrow j$
    end if
  end for
  $y^{(i)} \leftarrow \text{label}(u_{j*}^{(i)})$
  $\mathcal{L}^{(i+1)} \leftarrow \mathcal{L}^{(i)} \cup \{(u_{j*}^{(i)}, y^{(i)})\}$
  $\mathcal{U}^{(i+1)} \leftarrow \mathcal{U}^{(i)} \setminus \{u_{j*}^{(i)}\}$
end for
return $\frac{1}{B} \sum_{i=1}^{B} \text{acc}^{(i)}$

however we are solely focused on comparing different AL algorithms in this paper. Our core assumption is that a well-performing algorithm in our benchmark will transfer well into more practical use-cases.

Adapting the experimental setting from [3] we offer all our datasets in the raw setting as well as pre-encoded by a fixed embedding model that was trained in an unsupervised manner. The pre-encoded datasets offer multiple advantages in that they require smaller classification models (i.e. small MLPs) and in general much less sampled datapoints to archive close to upper bound performance (measured with the full dataset). They unify experimental setups for different datasets and therefore make them much more comparable.

4 Methodology

4.1 State Space

Version: 0.1

Since every AL agent needs a different state space our environment exposes a full state to the agent, so that the agent has full control of what information will be used.

The state can include the following information:

- The entire labeled dataset $\mathcal{L}^{(i)}$
- The entire unlabeled dataset $\mathcal{U}^{(i)}$
- A histogram of labeled points per class (count)
- The available budget B
- Number of added datapoints $|\mathcal{L}| - |\mathcal{S}|$
- The initial validation accuracy $\text{acc}^{(0)}$ and current validation accuracy $\text{acc}^{(i)}$
- The current classification model including all model weights $\theta^{(i)}$
- The current optimizer including it's full state
- The current sample of unlabeled points $u^{(i)}$

4.2 Training the Classifier

Generally, the classification model can be trained in two ways. Either you reset the parameters after each AL iteration and train the classifier from scratch with the updated labeled set $\mathcal{L}^{(i)}$, or you retain the previous state and only fine-tune the classifier on $\mathcal{L}^{(i)}$ for a reduced number of epochs. In this work we use the fine-tuning method for raw datasets to save computation, while we use the from-scratch training for embedded dataset, since they have very small classifiers and this method generally produces better results. Our fine-tuning scheme always trains for at least one epoch and employs an aggressive early stopping after that. The early stopping has patience 0, so it will stop as soon as the validation loss does no longer decrease.

4.3 Evaluation

Version: 0.1

We compare different AL algorithms based on their archived AUC score (Eq. 1). This score does incorporate performance in early stages (low budget) as well as capabilities to push the classifier in later stages (high budget). A good AL algorithm should be able to perform well in both scenarios.

TODO

5 Experiments

5.1 Splice Embedded

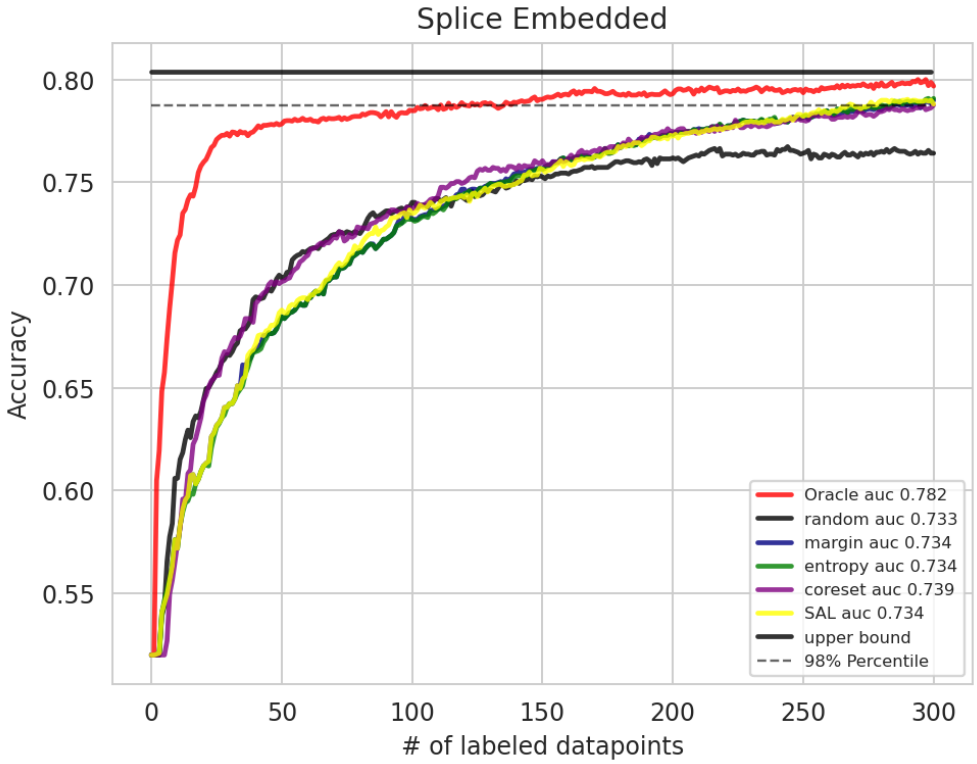

Figure 1: Results for all algorithms on the pre-encoded Splice dataset

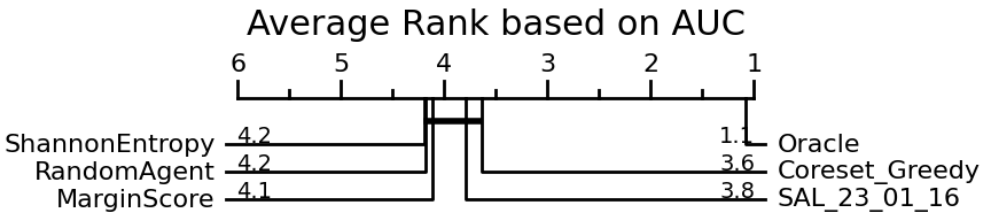

Figure 2: Critical Difference Diagram for Splice where every restart of the algorithm is one sample for the Wilcoxon-Holm method

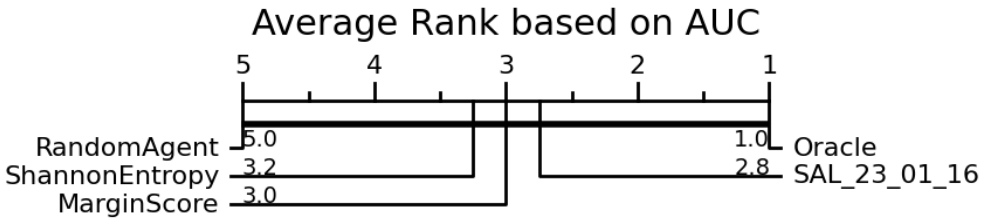

Figure 3: Critical Difference Diagram for Splice, DNA, USPS and Cifar10

6 Ablation Studies

- Setting τ to $|\mathcal{U}|$
- Reduction of the test set for speed

References

[1] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020.

[2] Nathan Beck, Durga Sivasubramanian, Apurva Dani, Ganesh Ramakrishnan, and Rishabh Iyer. Effective evaluation of deep active learning on image classification tasks. *arXiv preprint arXiv:2106.15324*, 2021.

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

[4] Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019.

[5] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[6] Yu Li, Muxi Chen, Yannan Liu, Daojing He, and Qiang Xu. An empirical study on the efficacy of deep active learning for image classification. *arXiv preprint arXiv:2212.03088*, 2022.

[7] Information Engineering Graduate Institute of Taiwan University. Libsvmtools.

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

[9] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

A Comparison of different sample sizes τ