Towards Comparable Active Learning

April 6, 2023

1 Introduction

1.1 Contribution

2 Related Work

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 [4] and TypiClust [2], 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 [3] and BADGE [1], which use metrics to measure the classifiers state.

Some previous work also aimed to provide a benchmark suite

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 $\mathcal{U} := u_i\,;\ i := 1 \dots N$ 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 4.3 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

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 $\text{Acc}(\mathcal{L}^{\text{test}}, \hat{y}_\theta)$

4 Methodology

4.1 Classification Model

The classifier is constructed according to two kinds of information. The general class of model (Dense, Convolutional, Attention, ...), and the configuration of the model (number of layers, size of each layer, ...). The model class and exact configuration is determined by the dataset, i.e. tabular datasets will prescribe a dense model. If special capabilities of the model are needed (i.e. Monte-Carlo Dropout), an extension of the given model class can be provided to the framework.

To ensure comparability between models, the model's configuration should not be changed or an additional evaluation of the new configuration should be conducted to compare the baseline expressivity.

4.2 State Space

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.

Algorithm 3 Oracle

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 $\frac{1}{B} \sum_{i=1}^{B} \text{acc}^{(i)}$

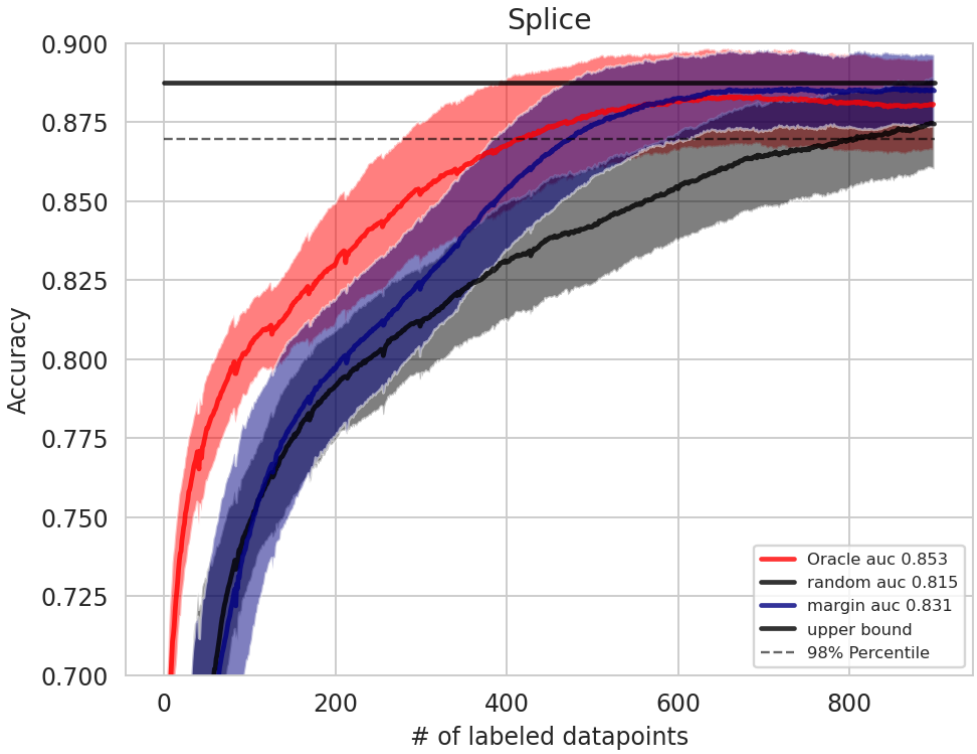

Figure 1: Performance drop for Oracle in late stages of Active Learning.

The state can include the following information:

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

4.3 Training the Classifier

4.4 Evaluation

5 Ablation Studies

- 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] 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.

[3] 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.

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

A Comparison of different sample sizes τ