Towards Comparable Active Learning

March 15, 2023

1 Introduction

1.1 Contribution

2 Related Work

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 $\hat{y}_\theta : \mathbb{R}^M \rightarrow \mathbb{R}^C$ is used. To fit the model, it is parameterized by θ and subjected to loss $\ell : \mathbb{R}^C \times \mathbb{R}^C \rightarrow \mathbb{R}$. 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: $\tau$                                           ▷ Unlabeled Sample Size
Require: $\Omega$                                         ▷ AL Agent
Require: $\omega$                                         ▷ Environment State function
 1: $\mathcal{L}^{(1)} \leftarrow \text{seed}(\mathcal{U})$     ▷ Create the initial labeled set
 2: $\mathcal{U}^{(1)} \leftarrow \mathcal{U}$
 3: for $i := 1 \dots B$ do
 4:     $\text{acc}^{(i)} \leftarrow \text{Retrain}(\mathcal{L}^{(i)})$   ▷ $\text{Retrain}(\mathcal{L}^{(i)})$ is shorthand for $\text{Retrain}(\mathcal{L}^{(i)}, \mathcal{L}^{\text{test}}, \hat{y}_\theta, e^{\max})$
 5:     $u^{(i)} \underset{\tau}{\sim} \mathcal{U}^{(i)}$
 6:     $s^{(i)} \leftarrow \omega(u^{(i)})$
 7:     $a^{(i)} \leftarrow \Omega(s^{(i)})$                 ▷ $a^{(i)}$ is an index inside of $u^{(i)}$
 8:     $y^{(i)} \leftarrow \text{label}(u_a^{(i)})$          ▷ $u_a^{(i)}$ is shorthand for $u_{a^{(i)}}^{(i)}$
 9:     $\mathcal{L}^{(i+1)} \leftarrow \mathcal{L}^{(i)} \cup \{(u_a^{(i)}, y^{(i)})\}$
10:     $\mathcal{U}^{(i+1)} \leftarrow \mathcal{U}^{(i)} \setminus \{u_a^{(i)}\}$
11: end for
12: 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
 1: $\text{loss}^* \leftarrow \infty$
 2: for $i := 1 \dots e^{\max}$ do
 3:     $\theta_{i+1} \leftarrow \theta_i - \eta \nabla_\theta \ell(\mathcal{L}, \hat{y}_\theta)$
 4:     $\text{loss}_i \leftarrow \ell(\mathcal{L}^{\text{test}}, \hat{y}_\theta)$
 5:     if $\text{loss}_i < \text{loss}^*$ then
 6:         $\text{loss}^* \leftarrow \text{loss}_i$
 7:     else
 8:         Break
 9:     end if
10: end for
11: return $\text{Acc}(\mathcal{L}^{\text{test}}, \hat{y}_\theta)$
```

Algorithm 3 Oracle

```
Require: $\mathcal{U}$                                    ▷ Unlabeled Pool
Require: $\tau$                                           ▷ Unlabeled Sample Size
Require: $\Omega$                                         ▷ AL Agent
Require: $\omega$                                         ▷ Environment State function
 1: $\mathcal{L}^{(1)} \leftarrow \text{seed}(\mathcal{U})$     ▷ Create the initial labeled set
 2: $\mathcal{U}^{(1)} \leftarrow \mathcal{U}$
 3: for $i := 1 \dots B$ do
 4:     $\text{acc}^{(0)} \leftarrow \text{Retrain}(\mathcal{L}^{(i)})$   ▷ $\text{Retrain}(\mathcal{L}^{(i)})$ is shorthand for $\text{Retrain}(\mathcal{L}^{(i)}, \mathcal{L}^{\text{test}}, \hat{y}_\theta, e^{\max})$
 5:     $u^{(i)} \underset{\tau}{\sim} \mathcal{U}^{(i)}$
 6:     $r* \leftarrow -\infty$
 7:     $j* \leftarrow -1$
 8:     for $j := 1 \dots \tau$ do
 9:         $y^{(j)} \leftarrow \text{label}(u_j^{(i)})$
10:         $\mathcal{L}^{(j)} \leftarrow \mathcal{L}^{(i)} \cup \{(u_j^{(i)}, y^{(j)})\}$
11:         $\mathcal{U}^{(j)} \leftarrow \mathcal{U}^{(i)} \setminus \{u_j^{(i)}\}$
12:         $\text{acc}^{(j)} \leftarrow \text{Retrain}(\mathcal{L}^{(j)})$
13:     end for
14:     $y^{(i)} \leftarrow \text{label}(u_j^{(i)})$
15:     $\mathcal{L}^{(i+1)} \leftarrow \mathcal{L}^{(i)} \cup \{(u_j^{(i)}, y^{(i)})\}$
16:     $\mathcal{U}^{(i+1)} \leftarrow \mathcal{U}^{(i)} \setminus \{u_j^{(i)}\}$
17:     $\text{acc}^{(i)} \leftarrow \text{Retrain}(\mathcal{L}^{(i+1)})$   ▷ $\text{Retrain}(\mathcal{L}^{(i+1)})$ is shorthand for $\text{Retrain}(\mathcal{L}^{(i+1)}, \mathcal{L}^{\text{test}}, \hat{y}_\theta, e^{\max})$
18: end for
19: return $\frac{1}{B}\sum_{i=1}^{B} \text{acc}^{(i)}$
```

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 callback-function that gives the agent full control over how the state is constructed.

The callback includes the following information:

- The current sample of IDs that point to the presented unlabeled datapoints
- 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

Every agent needs to implement this callback that transforms the given information into a state tensor that will be directly consumed by the agent to make it's prediction.

4.3 Training the Classifier

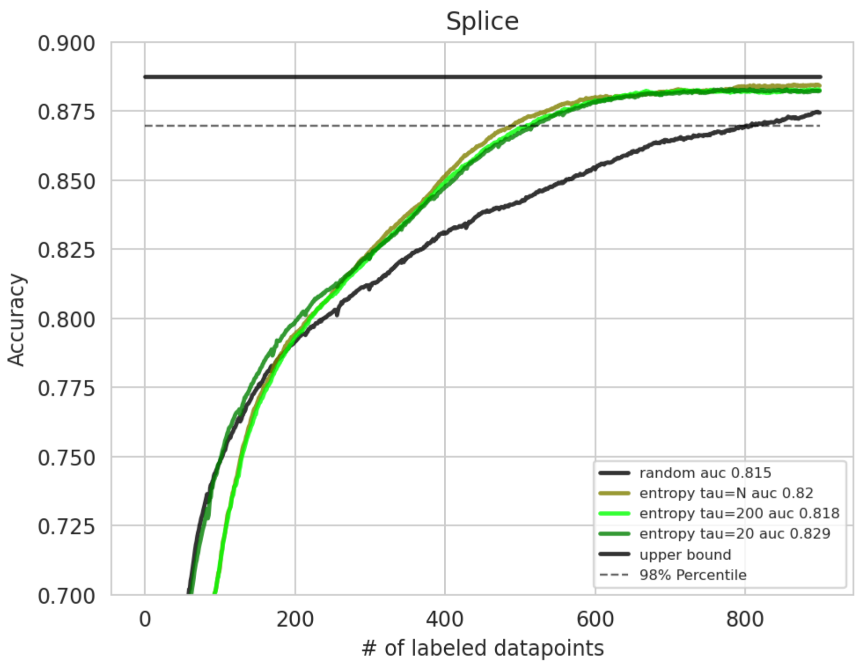
4.4 Evaluation

5 Ablation Studies

- Weird drop of performance for multiples of batch size (drop_last in DataLoader)
- Reduction of the test set for speed

References

A Comparison of different sample sizes τ


Splice
Accuracy
# of labeled datapoints
random auc 0.815
entropy tau=N auc 0.82
entropy tau=200 auc 0.818
entropy tau=20 auc 0.829
upper bound
98% Percentile