Towards Comparable Active Learning

February 16, 2023

1 Introduction

1.1 Contribution

- Benchmark suit for Active Learning
- Properly optimized classification models
- Multiple use-cases: Single Dataset, Dataset Transfer
- Multiple Domains: Tabular, Image, Text

2 Related Work

3 Overview

3.1 Problem Description

We assume a dataset $\mathcal{D} := (\mathcal{U}, \mathcal{L})$ consisting of a large pool of unlabeled data $\mathcal{U}$ and a small set of labeled data called the seed set with $|\mathcal{L}| \ll |\mathcal{U}|$. For the purpose of this work, only fully labeled datasets are used and the labels for $\mathcal{U}$ are suppressed until they are selected for labeling. For evaluation purposes we also assume a held-out test set $(x_{test}, y_{test}) \in (\mathbb{R}^M, \mathbb{R}^C)$ We consider only classification problems, hence the instances of a dataset have the form $x \in \mathbb{R}^M$ for $\mathcal{U}$ and $(x, y) \in (\mathbb{R}^M, \mathbb{R}^C)$ for $\mathcal{L}$. To perform classification, a model $\phi_\theta := \mathbb{R}^M \to \mathbb{R}^C$ is used. To fit the model, it is parameterized by θ and subjected to loss $l(y, \phi_\theta(x)) := \mathbb{R}^C \times \mathbb{R}^C \to \mathbb{R}$. For this work, categorical cross-entropy (CE) is used. For evaluating classification performance, we use Accuracy
Active learning is defined as sequentially selecting single instances of $x_i \in \mathcal{U}$, requesting their label y_i and adding them to the labeled pool $\mathcal{L} := \mathcal{L} \cup (x_i, y_i)$ until a fixed budget B is exhausted $i := 0 \ldots B$. After each added instance the classification model is retrained according to section 4.3 and its performance is measured on a held-out test set. 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 normalized AUC:

$$\text{auc}(\mathcal{U}, \mathcal{L}, \phi, B) := \frac{1}{B} \sum_{i=1}^{B} acc(y_{test}, \phi_i(x_{test})) \tag{1}$$

Where ϕ_i is the retrained classification model after the i-*th* instance was selected

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