Learning Active Learning from Real and Synthetic Data

Ksenia Konyushkova KSENIA.KONYUSHKOVA@EPFL.CH
EPFL

Raphael Sznitman RAPHAEL.SZNITMAN@ARTORG.UNIBE.CH
University of Bern

Pascal Fua PASCAL.FUA@EPFL.CH
EPFL

Abstract

In this paper, we suggest a novel data-driven approach to active learning: Learning Active Learning (LAL). The key idea behind LAL is to train a regressor that predicts the expected error reduction for a potential sample in a particular learning state. By treating the query selection procedure as a regression problem we are not restricted to dealing with existing AL heuristics; instead, we learn strategies based on experience from previous active learning experiments. We show that LAL can be learnt from a simple artificial 2D dataset and yields strategies that work well on real data from a wide range of domains. Moreover, if some domain-specific samples are available to bootstrap active learning, the LAL strategy can be tailored for a particular problem.

1. Introduction

Many modern machine learning techniques require large amounts of training data to reach their full potential. However, annotated data is hard and expensive to obtain, especially in specialized domains where only experts whose time is scarce and precious can provide reliable annotations. Active learning (AL) is an established way to ease the data collection process by automatically deciding which instances an annotator should label to train an algorithm as quickly as possible and with the minimal amount of annotation effort.

Over the years many AL strategies have been developed for various tasks, without any one of them clearly outperforming all others in all cases. Consequently, a number of approaches have been proposed to automatically select the best strategy. Recent examples include bandit algorithms (Baram et al., 2004; Hsu & Lin, 2015), aggregating strategies whose experience can be transferred between domains (Chu & Lin, 2016), and approaches based on Reinforcement Learning (Ebert et al., 2012).

A common limitation of all these approaches is that they cannot go beyond combining pre-existing, often hand-designed or computationally expensive, heuristics. In this paper, we propose to go further by allowing our method to discover the right strategies while being trained to discern effective AL approaches. In other words, our approach Learns Active Learning (LAL) and automatically comes up with task-appropriate query selection strategies.

More specifically, we formulate LAL as a regression problem. Given a trained classifier and its output for a specific sample with unknown label, we predict the reduction in generalization error that can be expected by adding the label to that point. In practice, we show that we can train such a regression function on synthetic data and generalize to the real data by taking features to be simple statistics, such as the variance of the classifier output or predicted probability distribution over possible labels for a specific datapoint. Furthermore, if one can provide a slightly larger initial set of annotated data, the regressor can be trained on it to help further extend the annotated set for very different classification tasks. We show that LAL works well on real data from several different domains such as biomedical imaging, economics, molecular biology, and high energy physics. It outperforms other methods without requiring hand-crafted heuristics.

2. Related Work

The last decade has produced many different AL strategies. They include uncertainty sampling (Tong & Koller, 2002; Joshi et al., 2009; Settles, 2010; Yang et al., 2015),

query-by-committee (Gilad-bachrach et al., 2005; Iglesias et al., 2011), expected model change (Settles, 2010; Sznitman & Jedynak, 2010; Vezhnevets et al., 2012), expected error (Joshi et al., 2012), and variance (Hoi et al., 2006) minimization, Bayesian AL (Houlsby et al., 2011). Among these, uncertainty sampling is both one of the simplest and one of the most popular. The intuition behind it is very natural as it proposes users to first label samples that are the most uncertain, that is, closest the classifier's current decision boundary. This strategy and most of the methods mentioned above work very well in cases such as the ones of Figs. 1, 2 but often fail in the more difficult one of Fig. 4. While both examples are synthetic, analogous situations arise regularly in real data (Baram et al., 2004).

In general there are two main types of AL strategies: theoretically motivated approaches that are not always tractable in real applications (*e.g.* expected reduction in error, information theoretic approaches) and approaches that demonstrated empirical increase in performance but whose convergence is not well understood. Among AL methods designed to handle complex real-world situations, some cater to specific classifiers, such as those that rely on Gaussian Processes (Kapoor et al., 2007), and applications, such as natural language processing (Tong & Koller, 2002; Olsson, 2009), sequence labeling tasks (Settles & Craven, 2008), visual recognition (Luo et al., 2004; Long et al., 2015), semantic segmentation (Vezhnevets et al., 2012), foreground-background segmentation (Konyushkova et al., 2015), image delineation (Mosinska et al., 2016), and preference learning (Singla et al., 2016). However, there is no one algorithm that outperforms all others for all tasks and datasets (Settles & Craven, 2008). Moreover, various querying strategies aim to maximize different performance metrics as evidenced in the case of multi-class classification (Settles, 2010).

As a result, meta learning algorithms have been gaining in popularity in recent years (Tamar et al., 2016), but they are rarely designed to deal with AL scenarios. The few existing approaches tackle the problem by combining AL strategies. Baram et al. (2004) combine several known heuristics during the AL execution with the help of a bandit algorithm. This is made possible by the use of the maximum entropy criterion, which estimates the classification performance without labels during AL. Hsu & Lin (2015) extend this line of work but move the focus from datasamples as arms to heuristics as arms in the bandit process and use a new unbiased estimator of the test error, which enables it to outperform the previous method. Chu & Lin (2016) go further and suggest transferring the bandit-learnt combination of AL heuristics between different tasks. Another view on the problem is presented in Ebert et al. (2012), where they balance exploration and exploitation with a Markov decision process.

The common limitation of these approaches is that they do not go beyond combining already existing approaches and are not designed to automatically propose new heuristics, as LAL does.

3. Towards Data Driven Active Learning

In this section we briefly introduce the standard Active Leaning (AL) framework along with uncertainty sampling (US), the most frequently-used heuristic to implement it (Tong & Koller, 2002; Joshi et al., 2009; Settles, 2010; Yang et al., 2015). Despite its simplicity, US often works remarkably well in practice. However, we will show that it displays an optimal behavior only in a limited number of situations and by this we will motivate the fact that one cannot rely on a single heuristic, no matter how good.

3.1. Active Learning (AL)

While the generic machine learning task is to find the best statistical model given training data, the task of AL is to select which data should be annotated in order to learn the model as quickly as possible. In practice, this means that instead of asking experts to annotate all the available data, we select iteratively which datapoints should be annotated next. The goal is to reach the a comparable accuracy level faster.

Suppose we are interested in classifying datapoints from a target dataset $\widetilde{\mathcal{D}} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where x_i is a d-dimensional feature representation of a datapoint and $y_i \in \{0, 1\}$ is its binary label. We choose a classifier f that can be trained on some $\mathcal{L}_t \subset \widetilde{\mathcal{D}}$ to map features to labels $f_{\mathcal{L}_t}(x_i) = \hat{y}_i$ through the predicted probability $p(y_i = y \mid \mathcal{L}_t, x_i)$. The standard AL procedure unfolds as follows.

1. It starts with a small labeled training dataset $\mathcal{L}_t \subset \widetilde{\mathcal{D}}$ with $t = 0$.

2. A classifier $f_{\mathcal{L}_t}$ is trained using $\mathcal{L}_t$.

3. $f_{\mathcal{L}_t}$ is applied to the remainder of the data $\mathcal{U}_t$ for which annotations are unavailable.

4. A query selection procedure picks an instance $x^* \in \mathcal{U}_t$ to be annotated at the next iteration. Usually, x^* maximizes a heuristic criterion based on the datasets $\mathcal{L}_t$ and $\mathcal{U}_t$, along with the predictions of the current classifier $p(y_i = y \mid \mathcal{L}_t, x_i)$ for unlabeled samples $x_i \in \mathcal{U}_t$.

5. x^* is given a label y^* by an expert, or an oracle in synthetic examples. We then take the new labeled set $\mathcal{L}_{t+1}$ to be $\mathcal{L}_t \cup (x^*, y^*)$ and the new sets of unlabeled samples to be $\mathcal{U}_{t+1} = \mathcal{U}_t \setminus x^*$.

6. Increment t, go back to step 2 and iterate until the desired accuracy is achieved or the number of iterations has reached a predefined limit.

3.2. Uncertainty Sampling

US has been reported to be successful in numerous scenarios and settings (Tong & Koller, 2002; Joshi et al., 2009; Settles, 2010; Yang et al., 2015; Konyushkova et al., 2015; Mosinska et al., 2016). As discussed in Section 2, it is based on selecting first samples about which the current classifier is least certain. There are many definitions of maximum uncertainty but one of the most widely accepted is to select samples that maximize the entropy $\mathcal{H}$ over predicted classes. In other words, this means taking x^* to be

$$\arg\max_{x_i \in \mathcal{U}_t} \mathcal{H}[p(y_i = y \mid x_i, \mathcal{L}_t)] \; . \tag{1}$$

3.3. Success, Failure, and Motivation

We now motivate our approach by presenting two toy examples, one in which US is empirically observed to be the optimal (greedy) solution and another one, where it makes suboptimal decisions. To this, let us consider a simple two-dimensional dataset $\widetilde{\mathcal{D}}$ that consists of two clouds of datapoints corresponding to two different classes: $\widetilde{\mathcal{D}} = \{(x_i, y_i)\}$, where $x_i \in \mathbb{R}^2$ and each class contains equal number of points, as shown in Fig. 1(a). The data in each cloud comes from a Gaussian distribution with a different mean but the same variance.

Let us further assume that a dataset $\widetilde{\mathcal{D}}'$ from the same distribution is available for testing. We can initialize the AL procedure of Section 3.1 given one sample from each class and their respective labels $\mathcal{L}_0 = \{(x_1, 0), (x_2, 1)\}$. $\mathcal{U}_0$ comprises the remaining data points. We start with a simple logistic regression classifier f on $\mathcal{L}_0$ and then test it on $\widetilde{\mathcal{D}}'$. If $\widetilde{\mathcal{D}}'$ is big, the error on it can be considered as a good approximation to the generalization error. The loss of $f_{\mathcal{L}_0}$ on $\widetilde{\mathcal{D}}'$ is $\ell_0 = \sum_{(x'_i, y'_i) \in \tilde{\mathcal{D}}'} \ell(\hat{y}_i, y'_i)$, where $\hat{y}_i = f_{\mathcal{L}_0}(x'_i)$. In this example we will consider simple $0/1$ loss.

Consider the US strategy that selects sample x^* according to Eq. 1. In this case it would be a datapoint with probability of class 0 close to 0.5. Let us try to label every point x from $\mathcal{U}_0$ one by one, form a new labeled set $\mathcal{L}_x = \mathcal{L}_0 \cup (x, y)$ and check what error a new classifier $f_{\mathcal{L}_x}$ yields on $\widetilde{\mathcal{D}}'$, that is, $\ell_x = \sum_{(x'_i, y'_i) \in \tilde{\mathcal{D}}'} (\hat{y}_i, y'_i)$, where $\hat{y}_i = f_{\mathcal{L}_x}(x'_i)$. The difference between errors obtained with classifiers constructed on $\mathcal{L}_0$ and $\mathcal{L}_x$ indicates how much the addition of a new datapoint x reduces the generalization error $\delta_\ell(x, \mathcal{L}_0) = \ell_0 - \ell_x$.

We repeat this experiment 10000 times and plot the mean (blue) and variance (red) of $\delta_\ell(x, \mathcal{L}_0)$ in Fig. 1(c). The points with p_i closest to 0.5, the one that US picks, is indeed close to being the one that yields the greatest error reduction.

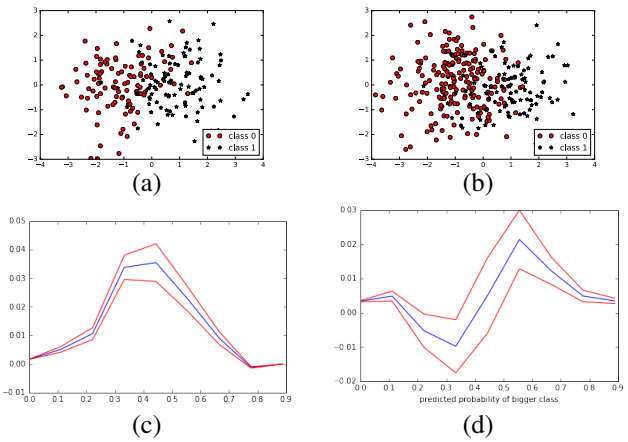

Figure 1. Balanced vs unbalanced. (a) Two Gaussian clouds of the same size. (b) Two Gaussian clouds with the class 0 being twice bigger than class 1. (c, d) The generalization error reduction in blue with associated variance in red as a function of predicted probability of class 0 in datasets (a) and (b), respectively.

Next, we repeat the above experiment, but with one class ($y = 0$) containing twice as many datapoints as the other ($y = 1$), as shown in Fig. 1(b). As before, we plot the average error reduction as a function of p_i in Fig. 1(d). However, this time the values of p_i that correspond to the largest expected error reduction are further from 0.5 than before and US becomes suboptimal. And the more unbalanced the two classes are, the more serious the bias becomes. In more complex and more realistic cases, there are many other factors such as label noise, outliers or shape of distribution that can further compound the problem,

This is why many query selection procedures take into account statistical properties of the datasets and classifier. However, there is no simple way to foresee the influence of all possible factors. Thus, in this paper, we suggest Learning Active Learning (LAL) by taking many factors into account to predict the potential error reduction. By treating the query selection module as a regressor we do not restrict ourselves to pre-existing AL heuristics but can construct new ones. This approach can be customized for any type of classifier and dataset. For instance, in the example of Section 3.3 we expect LAL to automatically adapt p_i to the relative prevalence of the two classes without having to explicitly state such a rule. We will see in the results section that this is indeed one of the many thing that LAL can handle, even in much more complex real-world cases.

Another AL scenario where learning AL strategy from data is useful is *warm start*. It is largely overlooked in the literature but has a significant practical interest, especially in highly specialized domains as we consider. We assume that

in order to understand if the machine learning based approach is possible in an application, some sufficient dataset $\mathcal{D}_0$ is made available prior to AL. However, further classification improvement can be very annotation costly. In this situation we can benefit from the available training data $\mathcal{D}_0$ to learn a specialized LAL strategy.

4. Approach

We now formulate LAL as a regression problem. We model the dependency between the state of the learning and the expected greedy improvement of the generalization error. To this end, given a representative dataset with ground truth available, we simulate an online learning procedure using a Monte-Carlo style approach. We further extend the approach to account for selection bias caused by AL.

More specifically, we split the dataset into training $\mathcal{D}$ and testing $\mathcal{D}'$ data and use a small subset of labels from $\mathcal{D}$ to train an initial classifier, which performance is tested on $\mathcal{D}'$. A number of parameters ϕ that characterizes the state of the classifier is evaluated. We then incorporate unused labels from $\mathcal{D}$ individually to retrain the classifier. This lets us correlate the increase in classification performance δ, or lack thereof, with the previously computed classifier parameters ϕ and parameters ψ of the newly added datapoint. We then repeat this process for different initializations, sizes of labeled subset, and initially selected samples. The iterative LAL strategy accounts for the bias introduce by AL into the training data and yields the best results as it will be shown in Sec. 6. We formalize this process in the remainder of the section.

4.1. Monte-Carlo LAL

Let the representative dataset discussed above be split into a training $\mathcal{D} = \{(x_i, y_i)\}$ and a testing set $\mathcal{D}' = \{(x'_i, y'_i)\}$ with $x_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$. Let f be a classifier with a given training procedure.

We start the LAL Monte-Carlo procedure by splitting $\mathcal{D}$ into a labeled set $\mathcal{L}_\tau$ of size τ and an unlabeled set $\mathcal{U}_\tau$ containing the remaining points. We then train a classifier f on $\mathcal{L}_\tau$, resulting in a function $f_{\mathcal{L}_\tau}$ that we use to predict class labels for elements x' of the testing set $\mathcal{D}'$ and estimate the test classification loss ℓ_τ. We further characterize the classifier state by recording k parameters $\phi_1(\mathcal{L}_\tau), \ldots, \phi_k(\mathcal{L}_\tau)$, which are specific to particular classifier families and change as the training progresses. For example, they can be the parameters of the kernel function if f is kernel-based, average depths of the trees if f is a random forest, or prediction variability if f is an ensemble classifier.

Next, we randomly select a new datapoint x from $\mathcal{U}_\tau$ to form a new labeled set $\mathcal{L}_{\tau+1} = \mathcal{L}_\tau \cup x$ and retrain. We characterize the effect of a datapoint on the state of learning by instantiating r parameters $\psi_1(x), \ldots, \psi_r(x)$. For example, they can include the predicted probability of class y, the distance to the closest point in the dataset or the distance to the closest labeled point. The new classifier $f_{\mathcal{L}_{\tau+1}}$ results in the test-set loss $\ell_{\tau+1}$. Finally, we record the difference between previous and new loss $\delta_\ell(\mathcal{L}_\tau, x) = \ell_\tau - \ell_{\tau+1}$. In the end, $\delta_\ell(\mathcal{L}_\tau, x)$ is associated to the learning state in which it was received. The learning state is characterized by a vector $\xi(\mathcal{L}_\tau, x) = \big[\phi_1(\mathcal{L}_\tau) \quad \ldots \quad \phi_k(\mathcal{L}_\tau) \quad \psi_1(x) \quad \ldots \quad \psi_r(x)\big]$, whose elements depend both on the state of the current classifier and on the datapoint, ϕ and ψ, respectively.

Next, we repeat the whole sampling procedure for Q different initializations $\mathcal{L}^1_\tau, \mathcal{L}^2_\tau, \ldots, \mathcal{L}^Q_\tau$ and T various labeled subset sizes $\tau = 2, \ldots, T+2$. For each initialization q and iteration τ we sample M different datapoints $x^{q\tau}_m$ each of which yields classifier/datapoint state pairs along with the reduction in error associated to them. This results in $Q \times M \times T$ MONTECARLOLAL observations $\xi_{q\tau m}$ of dimensionality $k + r$ that can be represented as a matrix Ξ whose lines are feature vectors:

$$\begin{bmatrix} \phi_1(\mathcal{L}^1_2) & \ldots & \phi_k(\mathcal{L}^1_2) & \psi_1(x^{12}_1) & \ldots & \psi_r(x^{12}_1) \\ \phi_1(\mathcal{L}^1_2) & \ldots & \phi_k(\mathcal{L}^1_2) & \psi_1(x^{12}_2) & \ldots & \psi_r(x^{12}_2) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathcal{L}^q_\tau) & \ldots & \phi_k(\mathcal{L}^q_\tau) & \psi_1(x^{q\tau}_m) & \ldots & \psi_r(x^{q\tau}_m) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathcal{L}^Q_T) & \ldots & \phi_k(\mathcal{L}^Q_T) & \psi_1(x^{QT}_M) & \ldots & \psi_r(x^{QT}_M) \end{bmatrix}$$

and associated vector Δ of labels $\delta_\ell(\mathcal{L}^q_\tau, x^m_{q\tau})$. Alg. 1 summarizes the steps of LAL data generation for a fixed τ and a given classifier f.

Our next insight is that these observations lie on a smooth manifold and that similar states of the classifier correspond to similar behaviors when annotating similar samples. We thus formulate the task of predicting the potential error reduction of annotating a specific sample in a given the classifier state as a regression problem. We look for a mapping

$$g : \xi(\mathcal{L}, x) \rightarrow \delta_\ell(\mathcal{L}, x). \tag{2}$$

Then MONTECARLOLAL strategy selects a datapoint with the highest error reduction potential at iteration t:

$$x^* = \underset{x_i \in \mathcal{U}_t}{\arg\max}\, g(\big[\phi_1(\mathcal{L}_t) \quad \ldots \quad \psi_r(x_i)\big]). \tag{3}$$

Note that MONTECARLOLAL does not depend explicitly on the representative dataset $\mathcal{D}$, but only on the chosen classifier and the choice of parameters ϕ and ψ. Therefore, it can be used to select samples that promise the greatest increase in classifier performance in domains $\widetilde{\mathcal{D}}$ *other* than the one we use to learn the strategy. Alg. 2 summarizes the MONTECARLOLAL learning procedure.

Algorithm 1 LALGENDATA

Input: training dataset $\mathcal{D}$, test dataset $\mathcal{D}'$, classification procedure f, partitioning function SPLIT, size τ
Initialize: $\mathcal{L}_\tau, \mathcal{U}_\tau \leftarrow$ SPLIT$(\mathcal{D}, \tau)$
train a classifier $f_{\mathcal{L}_\tau}$
estimate the test set loss ℓ_τ
compute the classification state parameters $\phi \leftarrow \phi_1(\mathcal{L}_\tau), \ldots, \phi_k(\mathcal{L}_\tau)$
for $m = 1$ **to** M **do**
  select $x_m \in \mathcal{U}_\tau$
  form a new labeled dataset $\mathcal{L}_{\tau+1} \leftarrow \mathcal{L}_\tau \cup x_m$
  compute the datapoint parameters $\psi \leftarrow \psi_1(x_m), \ldots, \psi_r(x_m)$
  train a classifier $f_{\mathcal{L}_{\tau+1}}$
  estimate the new test loss $\ell_{\tau+1}$
  compute the loss reduction $\delta_\ell(\mathcal{L}_\tau, x_m) \leftarrow \ell_\tau - \ell_{\tau+1}$
  $\xi_m \leftarrow \{\phi, \psi\}$
end for
$\Xi \leftarrow \{\xi_m\}$
Return: matrix of learning states $\Xi \in \mathbb{R}^{M \times (k+r)}$, vector of reductions in error $\Delta \in \mathbb{R}^{M \times 1}$

Algorithm 2 MONTECARLOLAL

Input: iteration range $\{\tau_{\min}, \ldots, \tau_{\max}\}$, classification procedure f
SPLIT $\leftarrow$ random partitioning function
Initialize: generate train set $\mathcal{D}$ and test dataset $\mathcal{D}'$
for τ **in** $\{\tau_{\min}, \ldots \tau_{\max}\}$ **do**
  $\Xi_\tau, \Delta_\tau \leftarrow$ LALGENDATA $(\mathcal{D}, \mathcal{D}', f,$ SPLIT$, \tau)$
end for
$\Xi, \Delta \leftarrow \{\Xi_\tau\}, \{\Delta_\tau\}$
train a regressor $g : \xi(\mathcal{L}, x) \rightarrow \delta_\ell(x, \mathcal{L})$ on data Ξ, Δ
construct AL strategy $\mathcal{A}(g)$:
$x^* = \arg\max_{x_i \in \mathcal{U}_t} g[\xi(\mathcal{L}_t, x_i)]$
Return: active learning strategy $\mathcal{A}(g)$

4.2. Iterative LAL

MONTECARLOLAL strategy of Alg. 2 makes an implicit over-simplification in the way the dataset $\mathcal{D}$ is split into a labeled set $\mathcal{L}_\tau$ and unlabeled set $\mathcal{U}_\tau$: No matter how many labeled samples τ are available, the labeled subset $\mathcal{L}_\tau$ consists of randomly chosen samples. However, when MONTECARLOLAL is applied at iteration t of AL, the labeled set $\mathcal{L}_t$ is constructed of samples selected by it at previous iterations, which is therefore *not* random.

To account for this, we modify the approach of Section 4.1 as follows. We learn a different AL strategy $\mathcal{A}_\tau$ at every iteration τ. $\mathcal{A}_\tau$ tells us which datapoint should be selected at iteration $t = \tau - 2$ of AL. When we execute iteration τ, we simulate AL procedure that start with 2 labeled samples and applies strategy $\mathcal{A}_2, \ldots, \mathcal{A}_\tau$ to select $3, \ldots, (\tau+1)$-th datapoints. In this way, samples at iteration $\tau + 1$ depend on the samples at iteration τ and the sampling bias of AL is well represented in data Ξ, Δ from which the final strategy is leant. Alg. 3 depicts the final procedure.

Algorithm 3 ITERATIVELAL

Input: iteration range $\{\tau_{\min}, \ldots \tau_{\max}\}$, classification procedure f
SPLIT $\leftarrow$ random partitioning function
Initialize: generate train set $\mathcal{D}$ and test dataset $\mathcal{D}'$
for τ **in** $\{\tau_{\min}, \ldots, \tau_{\max}\}$ **do**
  $\Xi_\tau, \Delta_\tau \leftarrow$ LALGENDATA $(\mathcal{D}, \mathcal{D}', f,$ SPLIT$, \tau)$
  train a regressor $g_\tau : \xi(\mathcal{L}, x) \rightarrow \delta_\ell(x, \mathcal{L})$ on Ξ_τ, Δ_τ
  SPLIT $\leftarrow \mathcal{A}(g_\tau)$
end for
$\Xi, \Delta \leftarrow \{\Xi_\tau, \Delta_\tau\}$
train a regressor $g : \xi(\mathcal{L}, x) \rightarrow \delta_\ell(x, \mathcal{L})$ on Ξ, Δ
Return: active learning strategy $\mathcal{A}(g)$

5. LAL Strategies Implementation

In this section we provide details about how we build the dataset Ξ, Δ to learn LAL strategies MONTECARLOLAL and ITERATIVELAL and select relevant features to instantiate the regression problem. We discuss the choice of classifier and regressor and provide values for key parameters for LAL procedure.

Representative data for training LAL. Recall from Section 4.1 that we learn the mapping g of Eq. 2 from Ξ, Δ that was obtained from an *example* data $\mathcal{D}$ that can be completely unrelated to the one on which we will do the AL $\widetilde{\mathcal{D}}$. Since the easiest way to obtain an example dataset is to synthesize it, we did this first and, as will be shown in the Results section, it turned out to be surprisingly effective.

In practice, we generate example 2-D datasets such as the ones depicted by Fig. 2 (a,b,c), where the 2 classes have Gaussian distributions with random means and variances and appear in various proportions. In our experiments, we set the size of training and test dataset to 400 and 4000 respectively and the imbalance between classes varies from 0.1 to 0.9. Each mean is drawn independently from a uniform distribution from 0 to 1 and the covariance is obtained by multiplying matrices whose entries are drawn uniformly between -0.5 and 0.5 with their transposes. We use $0/1$ loss through LAL data generation.

In *warm start* experiments, we used a small ($N_0 << N$) subset of data of the domain for AL, on which we learn $\mathcal{A}(g)$ and initial AL classifier $f_{\mathcal{L}_0}$. We split them into training and test subsets and ran the LAL procedures as described in Sec. 4.1 and Sec. 4.2.

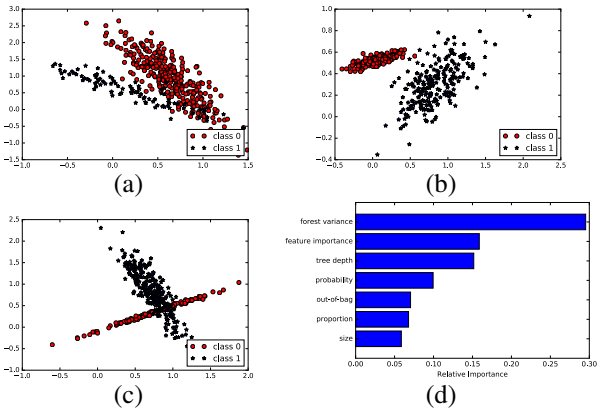

Figure 2. (a,b,c) Synthetic datasets used for LAL data collection. (d) Relative feature importance for **LAL-iterative-2D** using an RF regressor.

Choice of classifier and regressor. We use Random Forest (RF) and Gaussian Process (GP) classifiers for f and RF and GP regressors for g in the corresponding experiments. Due to the dimensionality of real datasets, GP was only used in experiment with simulated data Two-Gaussian-clouds.

Classifier features. The vector ξ that characterize the state of the learning process is composed of the following features: **a)** predicted *probability*: $p(y_i = 0|\mathcal{D}_t, x_i)$ **b)** *proportion* of class 0 in $\mathcal{D}_t$ **c)** *out-of-bag* cross-validated accuracy of $f_{\mathcal{D}_t}$ **d)** variance of *feature importances* of $f_{\mathcal{D}_t}$ **e)** *forest variance* computed as variance of trees' predictions on U_t **f)** average *tree depth* of the forest **g)** *size* of L_t.

Once the data Ξ, Δ was collected and regression solved, we can plot which features have the highest relative importance (Fig. 2 (d)).

In GP case we use **a)** predicted probability $p(y_i = 0|\mathcal{D}_t, x_i)$ **b)** predicted variance by GP **c)** variance and **d)** lengthscale of RBF kernel **e)** kernel density estimation for x_i with respect to labeled and **f)** unlabeles samples **g)** *size* of L_t.

Parameters used to construct LAL. The LAL data generation parameters are set to the following values: $M = 50$, $T = 48$, $Q = 500$. Moreover, for every new initialization we use a new representative dataset from model of Fig. 2 (a-c) that insures that the learnt strategy can generalize to various problems.

6. Experiments

We now compare the performance of LAL against several baselines both on synthetic and real data.

6.1. Baselines and Protocol

We compare the following three versions of our approach:

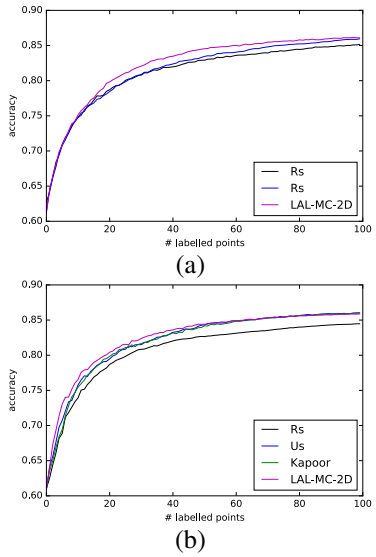

Figure 3. Two-clouds case. Accuracy on previously unseen datasets as a function of the number of labeled samples both for baselines and LAL strategies for (a) Random Forest classifier and (b) Gaussian Process classifier. LAL outperforms the baselines even when the performance is very close to the limit.

- **LAL-MC-2D**. LAL strategy learned as discussed in Sec. 4.1 where $\mathcal{D}$ is synthetic 2D dataset of Sec. 5.
- **LAL-iterative-2D**. Similar to **LAL-MC-2D** but learnt according to Sec. 4.2.
- **LAL-MC-WS**: Similar to **LAL-MC-2D** but $\mathcal{D}$ is a subset of the data $\widetilde{\mathcal{D}}$ on which AL is performed.

against the following baselines:

- **Rs**: random sampling.
- **Us**: uncertainty sampling according to Sec. 3.2.
- **Kapoor**: approach of Kapoor et al. (2007) that balances exploration against exploitation by incorporating mean and variance estimation of the GP classifier.

In our experiments we chose **Us** as our primary baseline because our approach makes use of the same information – predicted probability distribution over classes – as the datapoint features ψ. To compare against **Kapoor**, we included variance into ψ for a fair comparison.

In AL experiments we then select samples to be labeled from the training set and report the classification performance on the test set. When using **LAL-MC-2D** and **LAL-iterative-2D**, we begin with one sample from each of two classed to initialize AL. We will refer to this setting as a *cold start*. In **LAL-MC-WS** strategy we start with a larger training set, which we will refer to as a *warm start*.

Then, we perform from 100 to 1800 AL iterations depending on the dataset size and the task complexity and repeat

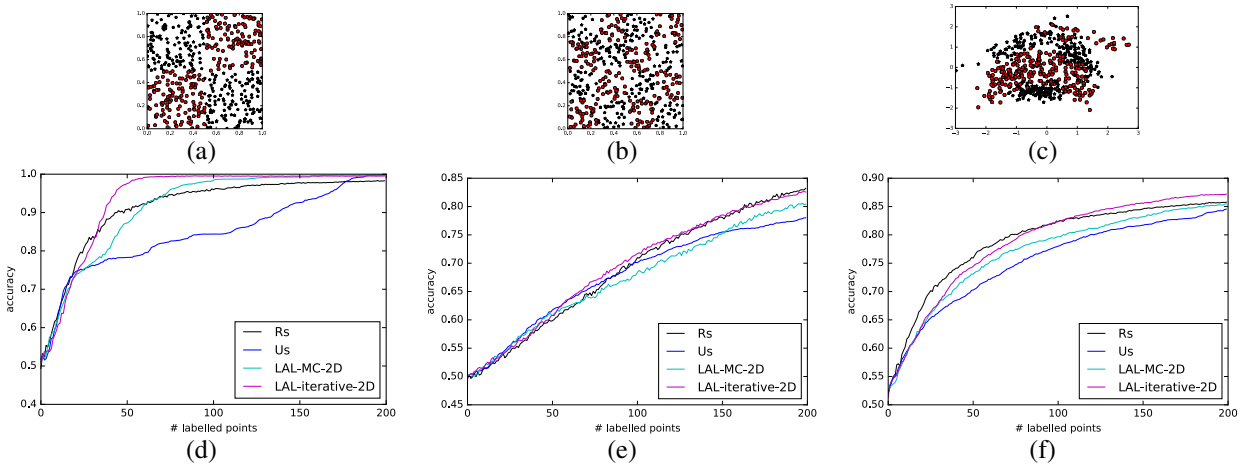

Figure 4. XOR-like case. (Top row) Three different datasets with one class depicted by red dots and the other by blue stars. (Botton row) Accuracy as a function of the number of labeled samples both for baselines and LAL strategies, In this case **Rs** is the best baseline and LAL outperforms it.

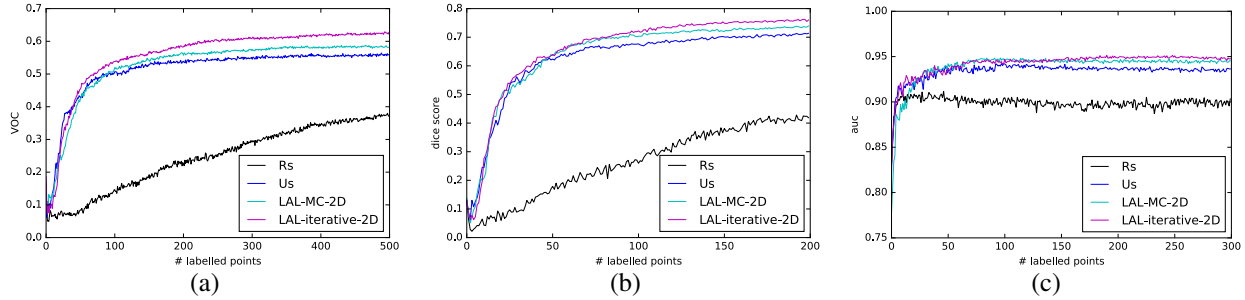

Figure 5. Task-dependent scores achieved by baselines and LAL strategies as a function of the number of labeled samples. (a) VOC on *striatum*. (b) dice on *MRI*. (c) AUC for *credit card*.

each experiment $50 - 100$ times. In the figures below, we plot average classification quality as a function of the number of samples having been annotated.

The performance metrics we use are task-specific, they include classification accuracy, VOC score (Everingham et al., 2010), dice score (Gordillo et al., 2013), AMS score (Adam-Bourdarios et al., 2014) and area under ROC curve.

6.2. Synthetic Data

We first demonstrate the performance of AL approach on the synthetic data generated using the model of Sec. 5, but previously unseen. Next we test our approach on the notoriously hard XOR-like datasets.

Two-Gaussian-clouds experiment. We generate 1000 new unseen datasets of the type depicted in Fig. 2 and test AL with GP and RF classifiers. Fig. 3 depicts the average test accuracy as a function of a number of labeled samples.

In case of both classifiers the proposed strategy is able to select datapoints that help to construct better classifiers quicker than **Us** heuristics. This experiment demonstrates that the LAL approach is robust to the type of chosen classifier f and parameters ϕ and ψ. Even in this simple task, when we are very close to the optimal performance, we can make better decisions by taking the learning state into account.

XOR-like experiment XOR-like datasets are well-known to be challenging for most machine learning techniques and AL is not an exception. Some works (Baram et al., 2004) report that various AL algorithms struggle with the type of tasks that is depicted in Fig. 4 (a), (b), (c). The reason for this is intuitively clear: any AL strategy relies on the prediction made by a classifier trained on available data. The prediction based on limited data is very distant from the ground truth in this case. When we have only two labeled instances in data like in Fig. 4 (a), at least two out of four groups of data are not represented in the training set at all. The classifier $f_{\mathcal{L}_2}$ will guide further selection to refine the initial boundary and completely overlook the other two groups of datapoints. The task for AL becomes even more challenging when the XOR structure resembles a checkerboard as depicted in Fig. 4 (b). In the limiting case like this

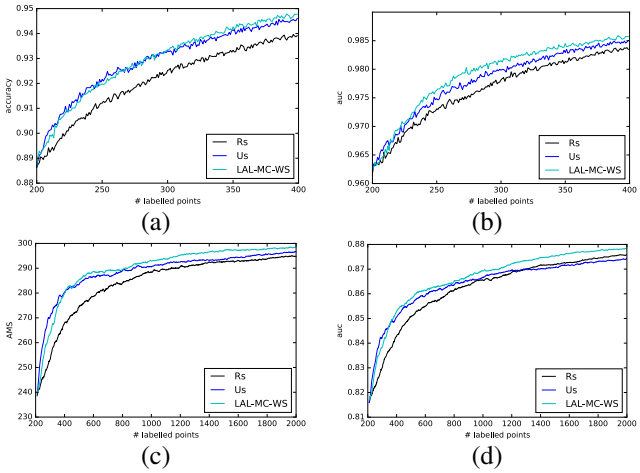

Figure 6. Warm start. (a,b) Accuracy and AUC scores for *Splice*. (c,d) AMS and AUC for Higgs dataset.

we cannot expect to get better quality of prediction faster than with random sampling. Another XOR-like dataset that has more natural shape is the banana dataset from Rätsch et al. (2001). Note that these datasets do not resemble in the least the data used to train LAL, as can be seen by comparing Figs.2 and 4).

Figs. 4(d), (e), (f) show the average performance of AL strategies for the classification tasks depicted by Figs. 4 (a), (b), (c). As expected, **Us** looses to **Rs**. Nevertheless, **LAL-iterative-2D** adapts its selection strategy to perform better or at least comparably to **Rs** for AL even under such adversarial conditions. In other words, even though LAL has never seen an XOR-like dataset during its training, it still performs well by adapting its selection strategy on the basis of the statistics of the samples it encounters.

6.3. Real Data

We now turn to real data from the domains where annotating is hard because it requires special training and education. Note that all the datasets are of a very different nature that means that their feature distributions have nothing to do with the syntactic example of Sec. 5.

- *Striatum*: 3D Electron Microscopy stack of rat neural tissue from striatum. The task is to detect and segment mitochondria in it (Lucchi et al., 2012; Konyushkova et al., 2015).
- *MRI*: brain scans with MRI are obtained from BRATS competition (Menza et al., 2014). The task is to segment brain tumor in T1, T2, FLAIR, and post-Gadolinium T1 MR images.
- *Credit card*: The task is to detect credit card fraud transactions in transaction made by European cardholders in September 2013 (Dal Pozzolo et al., 2015).
- *Splice*: In this dataset from the domain of molecular biology, our task is to detect splice junctions in DNA sequences (Lorena et al., 2002).
- *Higgs*: This dataset from the domain of high energy physics contains the data that simulates the ATLAS experiment (Adam-Bourdarios et al., 2014).

Cold Start AL. We start with a standard setup to benchmark AL algorithms. Fig. 5 depicts the results of applying **Rs**, **Us**, **LAL-MC-2D**, and **LAL-iterative-2D** in three datasets *Striatum, MRI, Credit card*. Recall that the LAL strategies still all rely on the same regressor learned from 2D synthetic data and have *not* been fine-tuned for the specific problems at hand.

Both LAL strategies outperform **Us**, with **LAL-iterative-2D** being the best of the two. Considering that the LAL regressor was learned using a simple 2D synthetic dataset, it is in fact somewhat surprising that it can deal with much more complex and unrelated datasets. This is something we intend to investigate further.

Warm Start AL. We now turn to a more realistic scenario where a larger dataset is available to train the initial classifier before beginning AL, as discussed in Section 3. We can take advantage of the larger initial training set to learn an AL strategy that is tailored for the problem at hand. This is valuable for applications where task or feature distribution is so different from synthetic data that AL strategy cannot be transfered, for example, for tasks where feature distributions contain missing or categorical variables. We tested **LAL-MC-WS** approach on the *Splice* and *Higgs* datasets by starting with 100 and 200 randomly selected datapoints. Fig. 6 demonstrates the learning curve for **Us** and for LAL. For *Splice* dataset we report accuracy and AUC and for *Higgs* dataset we report AMS score and AUC. We conclude that LAL approach is indeed useful in this problem settings.

7. Conclusion

In this paper we introduced a new approach to AL that is driven by data – Learning Active Learning. We found out that a LAL that was exposed to AL experiments on simple 2D data can sometimes generalize surprisingly well to challenging new domains. The ability to learn LAL from a subset of data of interest allows to further extend applicability of our approach. LAL demonstrated robustness to the choice of type of classifier and features.

In future work we would like to incorporate more features into LAL that will allow to compare it to various exploration/exploitation strategies. Furthermore we would like to address issues of multi-class classification and batch-mode AL.

Acknowledgements

This project has received funding from the European Unions Horizon 2020 Research and Innovation Programme under Grant Agreement No. 720270 (HBP SGA1). We would like to thank Carlos Becker, Helge Rhodin and Lucas Maystre for their discussions and comments on the text.

References

Adam-Bourdarios, Claire, Cowan, Glen, Germain, Cécile, Guyon, Isabelle, Kégl, Balázs, and Rousseau, David. The higgs boson machine learning challenge. In *NIPS 2014 Workshop on High-energy Physics and Machine Learning*, 2014.

Baram, Yoram, El-Yaniv, Ran, and K. Luz, Kobi. Online Choice of Active Learning Algorithms. *Journal of Machine Learning Research*, 5:255–291, 2004.

Chu, Hong-Min and Lin, Hsuan-Tien. Can active learning experience be transferred? 2016. URL http://arxiv.org/abs/1608.00667.

Dal Pozzolo, Andrea, Caelen, Olivier, Johnson, Reid A, and Bontempi, Gianluca. Calibrating probability with undersampling for unbalanced classification. In *Computational Intelligence, 2015 IEEE Symposium Series on*, pp. 159–166. IEEE, 2015.

Ebert, S., Fritz, M., and Schiele, B. RALF: A Reinforced Active Learning Formulation for Object Class Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Everingham, M., L. Van Gool and, C.K.I. Williams, Winn, J., and Zisserman, A. The Pascal Visual Object Classes Challenge (VOC2010) Results, 2010.

Gilad-bachrach, R., Navot, A., and Tishby, N. Query by Committee Made Real. In *Advances in Neural Information Processing Systems*, 2005.

Gordillo, N., Montseny, E., and Sobrevilla, P. State of the Art Survey on MRI Brain Tumor Segmentation. *Magnetic Resonance in Medicine*, 2013.

Hoi, S.C.and, Jin, R., Zhu, J., and Lyu, M.R. Batch Mode Active Learning and Its Application to Medical Image Classification. In *International Conference on Machine Learning*, 2006.

Houlsby, Neil, Huszár, Ferenc, Ghahramani, Zoubin, and Lengyel, Máté. Bayesian active learning for classification and preference learning. *stat*, 1050:24, 2011.

Hsu, Wei-Ning and Lin, Hsuan-Tien. Active learning by learning. *AAAI*, pp. 2659–2665, 2015.

Iglesias, J.E., Konukoglu, E., Montillo, A., Tu, Z., and Criminisi, A. Combining Generative and Discriminative Models for Semantic Segmentation. In *Information Processing in Medical Imaging*, 2011.

Joshi, A. J., Porikli, F., and Papanikolopoulos, N. P. Scalable Active Learning for Multiclass Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2259–2273, 2012.

Joshi, A.J., Porikli, F., and Papanikolopoulos, N. Multi-Class Active Learning for Image Classification. In *Conference on Computer Vision and Pattern Recognition*, 2009.

Kapoor, A., Grauman, K., Urtasun, R., and Darrell, T. Active Learning with Gaussian Processes for Object Categorization. In *International Conference on Computer Vision*, 2007.

Konyushkova, K., Sznitman, R., and Fua, P. Introducing Geometry into Active Learning for Image Segmentation. In *International Conference on Computer Vision*, 2015.

Long, J., Shelhamer, E., and Darrell, T. Fully Convolutional Networks for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2015.

Lorena, Ana Carolina, Batista, Gustavo EAPA, de Carvalho, André Carlos Ponce Leon Ferreira, and Monard, Maria Carolina. Splice junction recognition using machine learning techniques. In *WOB*, pp. 32–39, 2002.

Lucchi, A., Li, Y., Smith, K., and Fua, P. Structured Image Segmentation Using Kernelized Features. In *European Conference on Computer Vision*, pp. 400–413, October 2012.

Luo, T., Kramer, K., Samson, S., Remsen, A., Goldgof, D. B., Hall, L. O., and Hopkins, T. Active Learning to Recognize Multiple Types of Plankton. In *International Conference on Pattern Recognition*, 2004.

Menza, B., Jacas, A., et al. The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 2014.

Mosinska, A., Sznitman, R., Glowacki, P., and Fua, P. Active Learning for Delineation of Curvilinear Structures. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Olsson, F. A Literature Survey of Active Machine Learning in the Context of Natural Language Processing. *Swedish Institute of Computer Science*, 2009.

Rätsch, Gunnar, Onoda, Takashi, and Müller, K-R. Soft margins for adaboost. *Machine learning*, 42(3):287–320, 2001.

Settles, B. Active Learning Literature Survey. Technical report, University of Wisconsin–Madison, 2010.

Settles, B. and Craven, M. An Analysis of Active Learning Strategies for Sequence Labeling Tasks. In *Conference on Empirical Methods in Natural Language Processing*, pp. 1070–1079, 2008.

Singla, Adish, Tschiatschek, Sebastian, and Krause, Andreas. Actively learning hemimetrics with applications to eliciting user preferences. In *ICML*, pp. 412–420, 2016.

Sznitman, R. and Jedynak, B. Active Testing for Face Detection and Localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10):1914–1920, June 2010.

Tamar, Aviv, Levine, Sergey, Abbeel, Pieter, WU, YI, and Thomas, Garrett. Value iteration networks. In *Advances in Neural Information Processing Systems*, pp. 2146–2154, 2016.

Tong, S. and Koller, D. Support Vector Machine Active Learning with Applications to Text Classification. *Machine Learning*, 2002.

Vezhnevets, A., Ferrari, V., and Buhmann, J.M. Weakly Supervised Structured Output Learning for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2012.

Yang, Y., Ma, Z., Nie, F., Chang, X., and Hauptmann, A. G. Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization. *International Journal of Computer Vision*, 113(2):113–127, 2015.