Learning Active Learning from Data

Computer Vision Laboratory (CVLAB)

Ksenia KONYUSHKOVA, Raphael SZNITMAN, Pascal FUA

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

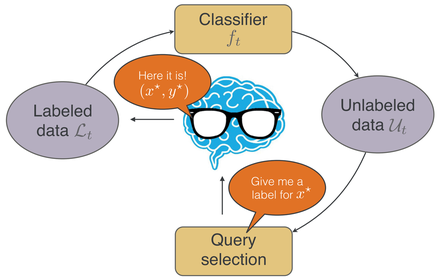
Active Learning

Can machines learn faster?

Many modern machine learning techniques require large amounts of training data to reach their full potential.

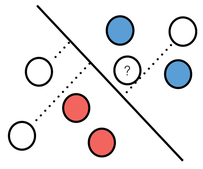
However, annotated data is hard and expensive to obtain, notably in specialized domains.

Active learning aims to ease the data collection process by deciding which instances should be labeled to train an algorithm as quickly and effectively as possible.



Uncertainty sampling:
query the most uncertain sample

$$x^\star = \arg\max_{x \in \mathcal{U}_t} H(\hat{y} = y \mid x))$$



Over the years many AL strategies have been developed for various classification tasks, without any one of them clearly winning in all cases.

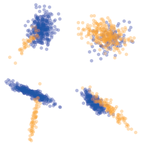
Solution: automatically select the best strategy.

Existing methods: combine hand-designed heuristics from a pre-defined set.

In our work:

1) a whole *continuum* of AL strategies

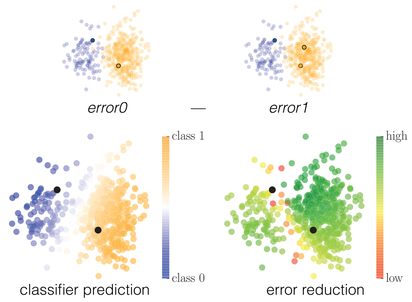
2) *experience* from previous tasks allows to transfer strategies to new domains



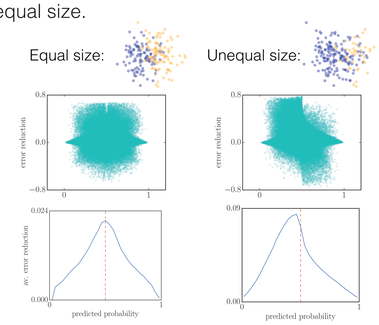
Success, Failure and Motivation

Which active learning strategy is the best?

Let us consider an example where we start with one point from each class and try to add a third datapoint.



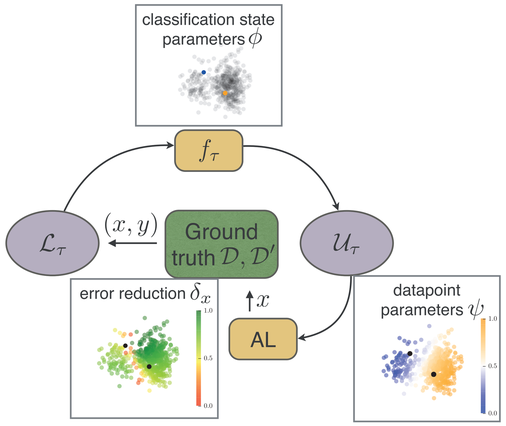
We repeat the experiment 1000 times and look at the average error reduction when classes are of equal or not equal size.



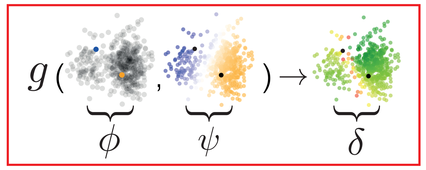
Learning Active Leanring (LAL)

What if we could learn an active learning strategy?

Learning Active Learning (LAL) uses properties of classifiers and data to predict the potential error reduction. Given a dataset with ground truth, we simulate an online learning procedure using a Monte-Carlo approach.



We treat the query selection problem as a regression model:



As a classifier we use Random Forest classifier. The state of the learning process consists of:

- predicted probability
- proportions of the classes
- cross-validated accuracy
- feature importances
- forest variance
- tree depth
- size of labeled set

and many other parameters can be included depending on the dataset and classification model!

DATAMONTECARLO

Input: training and test datasets $\mathcal{D}$, $\mathcal{D}'$,
classification procedure f,
partitioning function SPLIT,
size τ
Initialize: $\mathcal{L}_\tau, \mathcal{U}_\tau \leftarrow$ SPLIT$(\mathcal{D}, \tau)$
train a classifier f_τ
estimate the test set loss ℓ_τ
compute the classification state parameters ϕ
for $m = 1$ **to** M **do**
  select $x \in \mathcal{U}_\tau$
  form a new labeled dataset $\mathcal{L}_x \leftarrow \mathcal{L}_\tau \cup \{x\}$
  compute the datapoint parameters ψ
  train a classifier f_x
  estimate the new test loss ℓ_x
  compute the loss reduction $\delta_x \leftarrow \ell_\tau - \ell_x$
  $\xi_m \leftarrow \{\phi, \psi\}$, $\delta_m \leftarrow \delta_x$
$\Xi \leftarrow \{\xi_m\}$, $\Delta \leftarrow \{\delta_m\}$
Return: matrix of learning states Ξ, vector of reductions in error Δ

LALINDEPENDENT
incorporates labels one by one to lear a regressor

BUILDLALINDEPENDENT

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**
  for $q = 1$ **to** Q **do**
    $\Xi_{\tau q}, \Delta_{\tau q} \leftarrow$ DATAMONTECARLO $(\mathcal{D}, \mathcal{D}', f, \text{SPLIT}, \tau)$
$\Xi, \Delta \leftarrow \{\Xi_{\tau q}\}, \{\Delta_{\tau q}\}$
train a regressor $g : \xi \to \delta$ on data Ξ, Δ
construct LALINDEPENDENT $\mathcal{A}(g)$:
$x^* = \arg\max_{x \in \mathcal{U}_t} g[\xi_{t,x})]$
Return: LALINDEPENDENT

LALITERATIVE
accounts for selection bias caused by AL

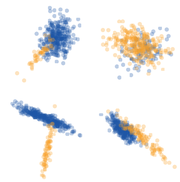
BUILDLALITERATIVE

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**
  for $q = 1$ **to** Q **do**
    $\Xi_{\tau q}, \Delta_{\tau q} \leftarrow$ DATAMONTECARLO $(\mathcal{D}, \mathcal{D}', f, \text{SPLIT}, \tau)$
  $\Xi_\tau, \Delta_\tau \leftarrow \{\Xi_{\tau q}, \Delta_{\tau q}\}$
  train regressor $g_\tau : \xi \to \delta$ on Ξ_τ, Δ_τ
  SPLIT $\leftarrow \mathcal{A}(g_\tau)$
$\Xi, \Delta \leftarrow \{\Xi_\tau, \Delta_\tau\}$
train a regressor $g : \xi \to \delta$ on Ξ, Δ
construct LALITERATIVE $\mathcal{A}(g)$:
Return: LALITERATIVE

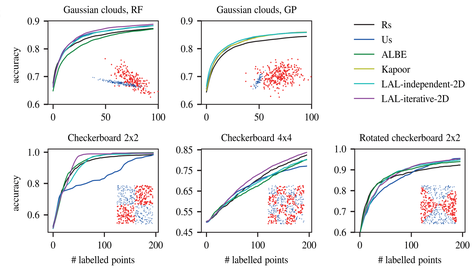
Experiments

Cold start

We start with one sample from each class, LAL is learnt on synthetic data.



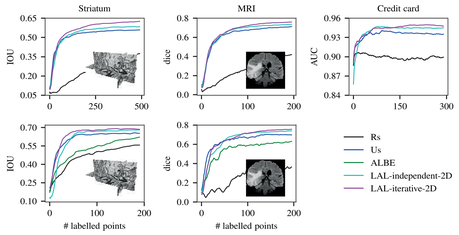
Artificial datastes



Baselines:

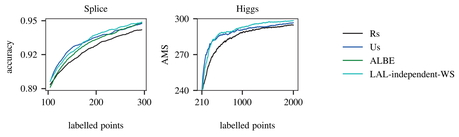
- **Rs**: random sampling
- **Us**: uncertainty sampling
- **Kapoor**: Kapoor et al., Active Learning with Gaussian Processes for Object Categorization
- **ALBE**: a recent example of meta-AL, Hsu and Lin, Active learning by learning

Real datastes



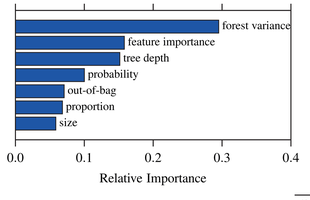
Warm start, real datasets

Larger dataset is available prior to AL, we can benefit from the available training data to learn a specialized AL strategy for an application

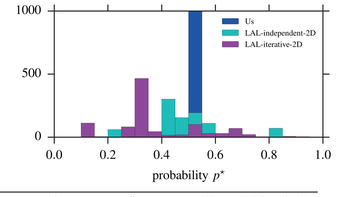


Analysing the LAL strategies

Understanding regressor: feature importances of the RF regressor representing strategy



Understanding selection: histograms of the selected probability for different AL strategies in experiments with MRI dataset



Computational cost
Time in seconds for one iteration of AL for various strategies and tasks

Dataset	Dimensions	# samples	Us	ALBE	LAL
Checkerboard	2	1000	0.11	13.12	0.54
MRI mini	188	2000	0.11	64.52	0.55
MRI	188	22 934	0.12	—	0.88
Striatum mini	272	2000	0.11	75.64	0.59
Striatum	272	276 130	2.05	—	19.50
Credit	30	142 404	0.43	—	4.73

Other information

The code is available at *https://github.com/ksenia-konyushkova/LAL*

The website of the author: *http://ksenia.konyushkova.com*

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 720270 (HBP SGA1)