Multi-Domain Active Learning for Semi-Supervised Anomaly Detection

Vincent Vercruyssen
Lorenzo Perini
Wannes Meert
Jesse Davis

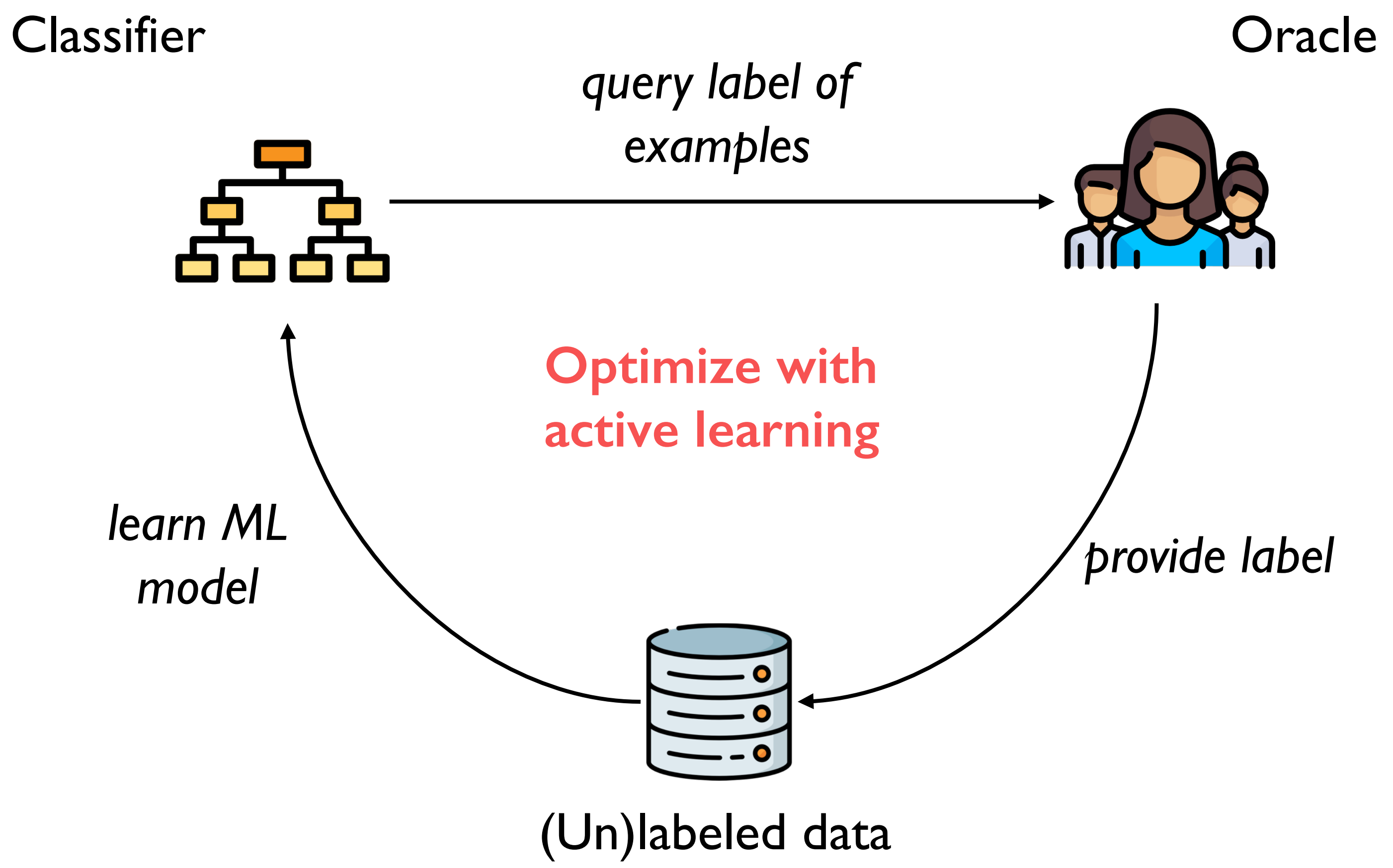
DTAI Research Group, KU LEUVEN, Belgium

vincent.vercruyssen@kuleuven.be

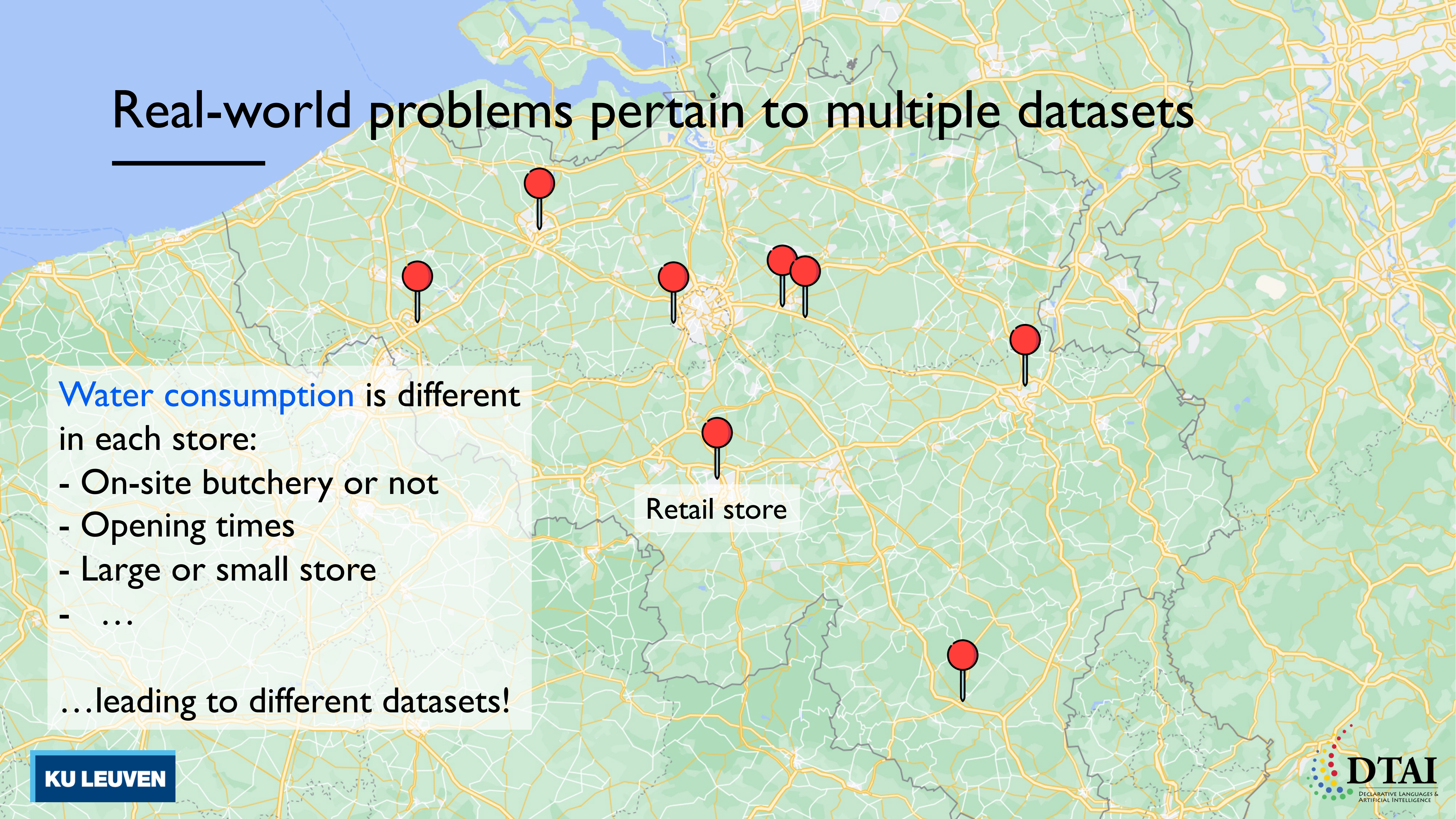
@VercruyssenV

KU LEUVEN   LEUVEN.AI   Spottr   DTAI Declarative Languages & Artificial Intelligence

Active learning



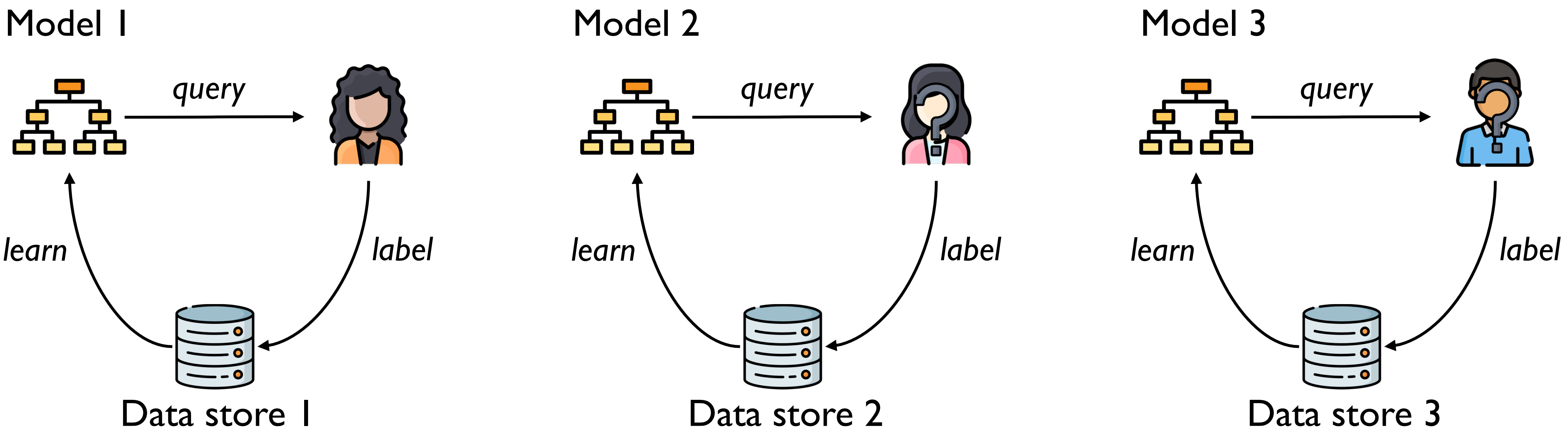
Real-world problems pertain to multiple datasets



Water consumption is different in each store:
- On-site butchery or not
- Opening times
- Large or small store
- …

…leading to different datasets!

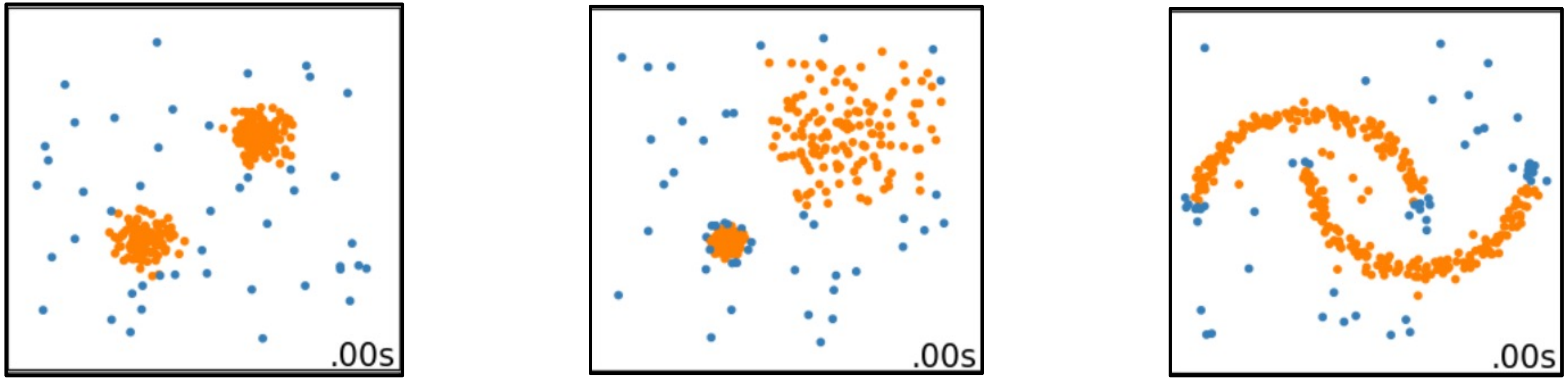
Different datasets necessitate different models



What if we only have 1 expert that has limited time to answer queries?

1) Some datasets require more expert labels than others

Labeling payoff = marginal gain of acquiring another label for a given dataset



LOW labeling payoff

HIGH labeling payoff

2) Active learning entails diminishing labeling payoff

The more labels given by the oracle for a dataset, the less the marginal gain of providing an additional label

Multi-domain active learning

GIVEN: a multi-domain dataset M consisting of K datasets, a fixed query budget T, and an expert oracle O

DO: learn a classifier for each of the K datasets with active learning

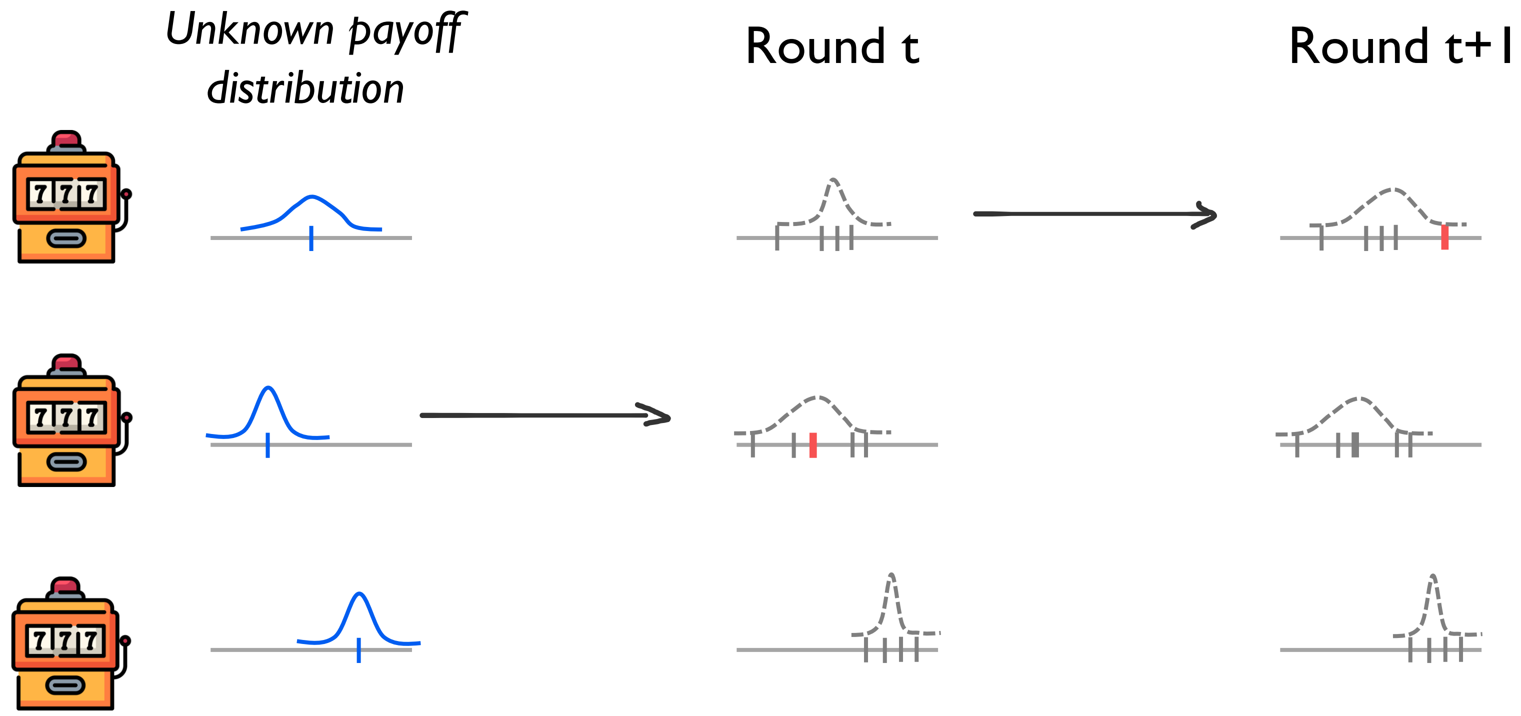
Pool-based active learning with a query batch size of one

Central to this problem is the exploration versus exploitation trade-off

exploration = figuring out which datasets are most useful to label

exploitation = providing the most labels in datasets that need it most

Exploration-exploitation with multi-armed bandits (MAB)



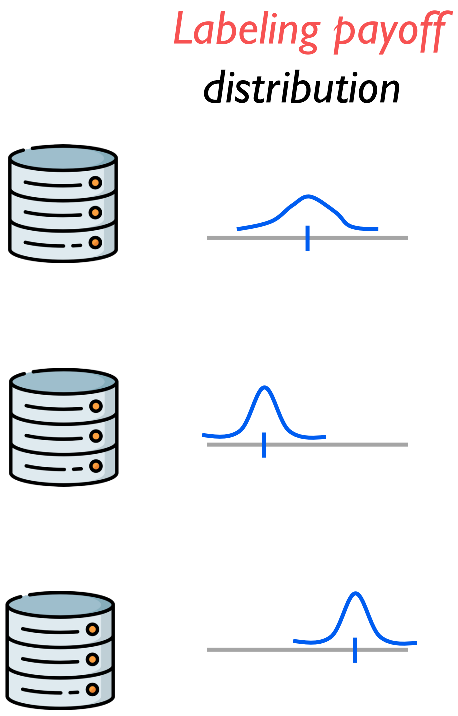
1) choose arm

2) observe reward/payoff

3) estimate reward distribution

More exploration ⟷ More exploitation

An MAB algorithm to tackle multi-domain active learning



Solving active learning via a multi-armed bandit approach is challenging:

1. How to equate groups of examples with the *arms* that the MAB can choose from?
2. How to select an *individual example* to query?
3. How do we quantify the *reward* for asking a query and getting the label?
4. How do we deal with the *diminishing returns* of labeling an additional example?

1. An MAB algorithm chooses between *arms*

Each arm corresponds to selecting a *group of examples* from which to query one

1. Arm = one of the K *datasets* to query an example from

   $$|arms| = K$$

   ✓ less exploration needed
   ✗ less fine-grained control

2. Arm = a *cluster in a dataset* to query an example from

   $$|arms| = K \times C$$

   ✗ more exploration needed
   ✓ more fine-grained control

2. Randomly select the final query example from a group

The MAB algorithm chooses one arm at each iteration

Each arm corresponds to selecting a *group of examples*

1. Select an example in the group *randomly*

   ✗ not necessarily query most informative examples

   ✓ unbiased estimate of the reward distribution

2. Select an example in the group *heuristically*

   ✓ select examples with highest estimated labeling payoff

   ✗ distorts the reward distribution estimate

Reasoning in the paper!

3. Playing an arm results in an observed *reward*

The reward of each action taken by the MAB algorithm reflects the labeling payoff

1. Reward = *entropy reduction* in the predictions of the underlying classifier

$$r = \sum_{x \in D^k} \left[H_{f_+^k}(x) - H_{f^k}(x) \right]$$

2. Reward = *number of examples* for which the prediction of the classifier changes

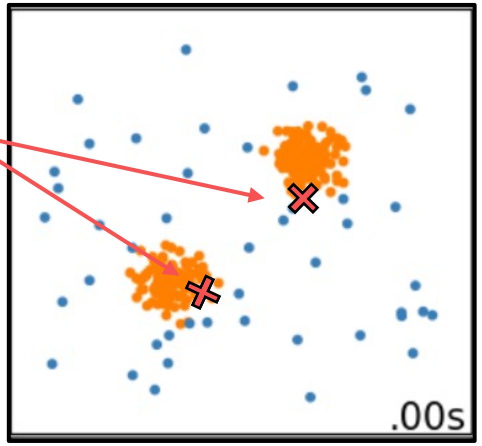
$$r = 1 - \frac{Y_{f_+^k} \cdot Y_{f^k}}{\left\| Y_{f_+^k} \right\| \left\| Y_{f^k} \right\|}$$

4. Model diminishing labeling payoff with *rotting bandits*

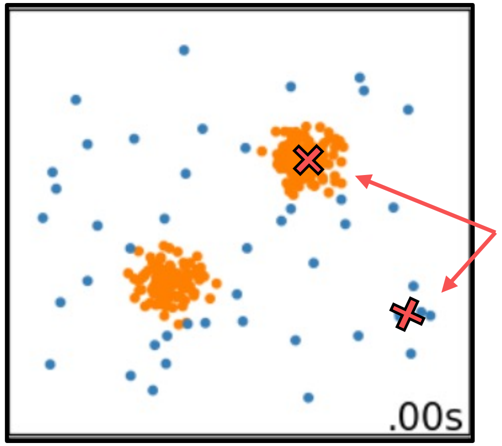
The more labels given by the oracle for a dataset, the less the marginal gain of providing an additional label

Initial label iterations

High-impact labels



Final label iterations



Low-impact labels

ALBA uses the *Sliding Window Average* (SWA) rotting bandit algorithm

Levine, N., Crammer, K., and Mannor, S. (2017). Rotting bandits. In: *Advances in Neural Information Processing Systems* 30.

Our *Active Learning Bandits* (ALBA) algorithm in full

```
Input: multi-domain dataset M, budget T, oracle O, number of clusters C
Output: set of trained classifiers

1. A ← divide each dataset ∈ M into C clusters
2. F ← Train an initial classifier for each dataset ∈ M
3. WHILE t < T:
4.     k ← SWA picks a cluster ∈ A based on the reward distribution
5.     x ← randomly select example in cluster k
6.     l ← query x and receive label from O
7.     update the classifier for the corresponding dataset
8.     compute reward and update cluster k's reward distribution
9.     t = t + 1
```

KU LEUVEN

DTAI
Declarative Languages & Artificial Intelligence

Differences between ALBA and active learning

Classic active learning

1. Estimates labeling payoff of every *single* example

2. Estimates labeling payoff of an example *before* it is queried

3. Works for a single dataset with a single classifier

Active learning Bandits (ALBA)

1. Estimates labeling payoff of *groups* of examples

2. Directly observes true payoff *after* an example is queried

3. Works for multiple datasets, each with its own classifier

Experimental evaluation

Our paper addresses three research questions:

- **Q1**: does ALBA outperform the existing active learning baselines for multi-domain active learning

- **Q2**: how does the division in groups of examples impact ALBA's performance

- **Q3**: how do the choice of reward function and query selection strategy impact ALBA's performance

Benchmark based on retail water consumption data

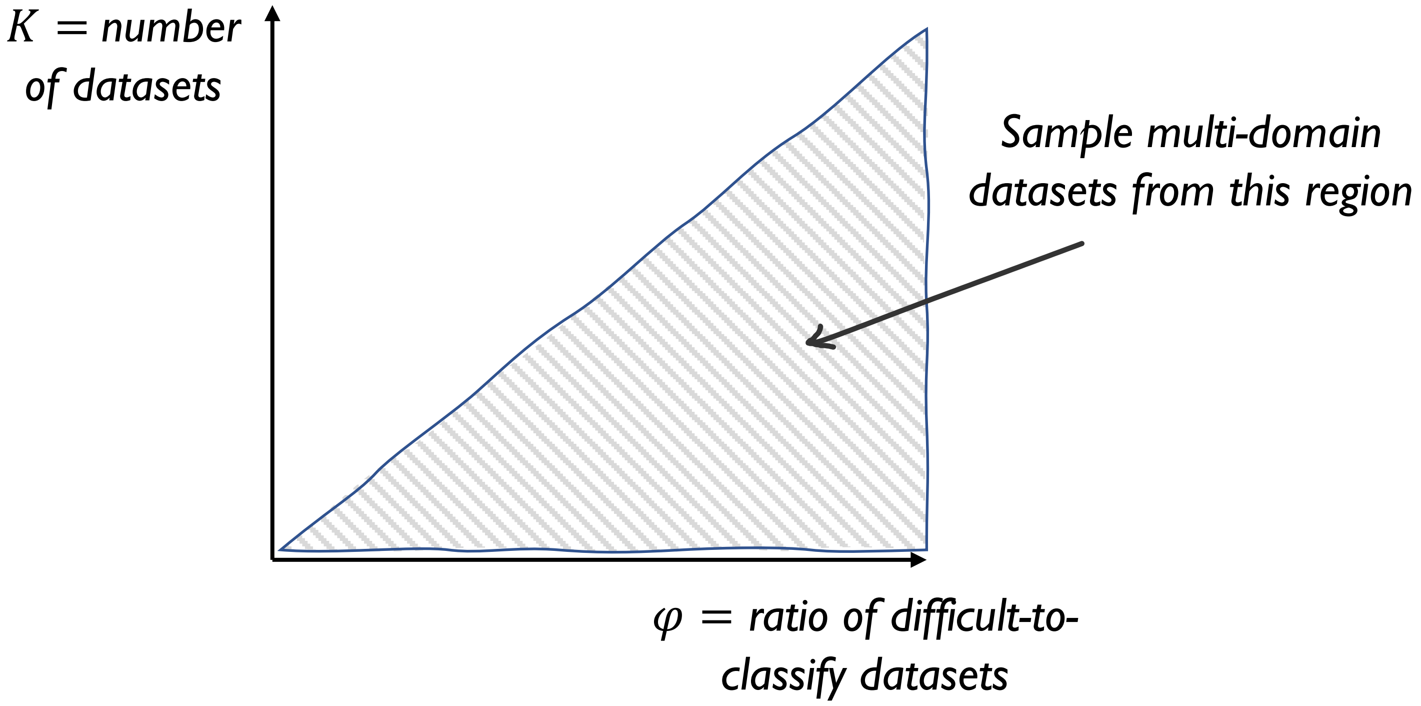
Several years of water consumption data for 7 retail stores

Task = classifying periods of abnormal water consumption

Final benchmark of 54 multi-domain datasets

Benchmark based on retail water consumption data

From the 7 datasets, we constructed a full benchmark of 54 multi-domain datasets



Baselines and evaluation

7 baselines

Combine all datasets into 1 dataset and learn a single classifier:

- C-RAND: acquire new labels randomly
- C-UC: acquire new labels heuristically

Treat each dataset independently and learn a separate classifier for each dataset:

- I-U: acquire no labels
- I-RAND: acquire new labels randomly
- I-UC: acquire new labels heuristically
- I-R-RAND: I-RAND + max queries / dataset
- I-R-UC: I-UC + max queries / dataset

Evaluation

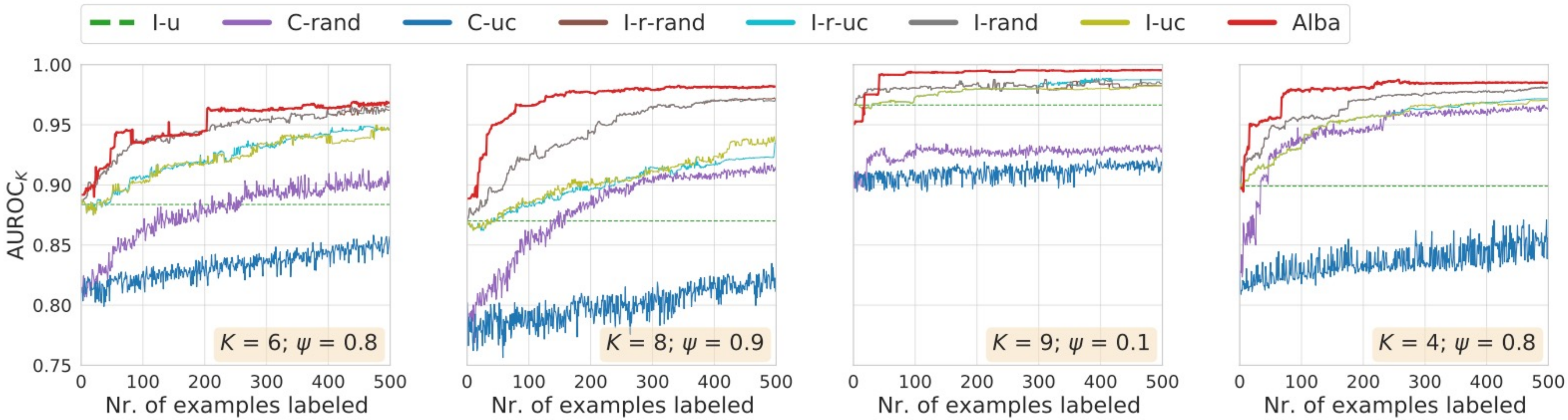
Area under the ROC curve measures the base performance of each anomaly classifier (we use the SSDO detector)

AUROC averaged over the K classifiers

The learning curve plots the AUROC as a function of the number of queried & labeled examples

Area under the learning curve (AULC) measures the impact of the active learning strategy (higher = better)

Q1: ALBA outperforms the active learning baselines (1/2)



Friedman test: not all methods perform similarly

Bonferroni-Dunn test: ALBA is significantly better @ 100 query rounds

Q1: ALBA outperforms the active learning baselines (2/2)

	Nr. of times ALBA:			Ranking
Method	wins	draws	loses	Avg. ± SD
ALBA	-	-	-	**1.315 ± 0.894**
I-RAND	**48**	2	4	2.639 ± 0.573
I-R-RAND	**48**	2	4	2.639 ± 0.573
I-U	**48**	2	4	4.333 ± 1.656
I-UC	**53**	0	1	5.157 ± 0.551
I-R-UC	**53**	0	1	5.231 ± 0.497
C-RAND	**54**	0	0	6.741 ± 0.865
C-UC	**54**	0	0	7.944 ± 0.404

(a) Results @ 100 query rounds

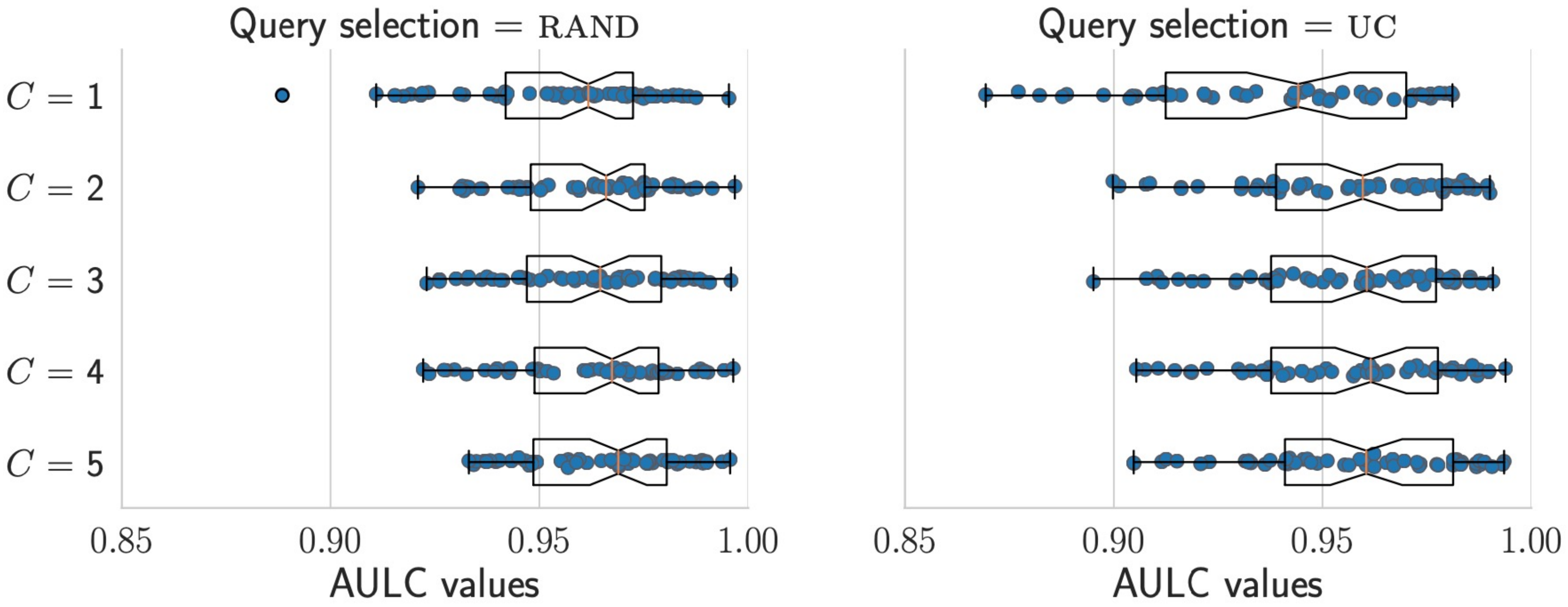
	Nr. of times ALBA:			Ranking
Method	wins	draws	loses	Avg. ± SD
ALBA	-	-	-	**1.306 ± 0.710**
I-RAND	**45**	2	7	2.417 ± 0.507
I-R-RAND	**46**	1	7	2.417 ± 0.507
I-R-UC	**54**	0	0	4.537 ± 0.686
I-UC	**53**	1	0	4.546 ± 0.512
I-U	**54**	0	0	6.370 ± 0.818
C-RAND	**54**	0	0	6.491 ± 0.717
C-UC	**54**	0	0	7.917 ± 0.382

(b) Results @ 500 query rounds

Friedman test: not all methods perform similarly

Bonferroni-Dunn test: ALBA is significantly better @ 100 query rounds

Q2: More clusters improves ALBA's performance



Correlation: higher C = higher ALBA performance

Q3: cosine reward function + random example selection

Reward function	Query sel. strategy	Ranking Avg. ± SD
cosine	rand	**1.806 ± 0.813**
cosine	uc	2.944 ± 0.926
entropy	rand	2.241 ± 0.843
entropy	uc	3.009 ± 0.825

Reward function: cosine > entropy

Example selection strategy: random > heuristic (uncertainty sampling)

Conclusions

1. Multi-armed bandit strategies are an effective tool for multi-domain active learning
2. Our contribution, the ALBA algorithm, outperforms existing AL algorithms
3. Constructing a large benchmark helped us gain insight in the performance

Feel free to ask any question!

Vincent Vercruyssen, *Lorenzo Perini, Wannes Meert, Jesse Davis*

https://github.com/Vincent-Vercruyssen/ALBA-paper

KU LEUVEN

DTAI
Declarative Languages & Artificial Intelligence