Daniel Kottke Vincent Lemaire Adrian Calma
Georg Krempl Andreas Holzinger

IAL@ECML PKDD 2019

Workshop & Tutorial on Interactive Adaptive Learning

Proceedings

The European Conference on Machine Learning and
Principles and Practice of Knowledge Discovery in Databases
(ECML PKDD 2019)

Würzburg, Germany, September 16, 2019

Preface

Science, technology, and commerce increasingly recognise the importance of machine learning approaches for data-intensive, evidence-based decision making. This is accompanied by increasing numbers of machine learning applications and volumes of data. Nevertheless, the capacities of processing systems or human supervisors or domain experts remain limited in real-world applications. Furthermore, many applications require fast reaction to new situations, which means that first predictive models need to be available even if little data is yet available. Therefore approaches are needed that optimise the whole learning process, including the interaction with human supervisors, processing systems, and data of various kind and at different timings: techniques for estimating the impact of additional resources (e.g. data) on the learning progress; techniques for the active selection of the information processed or queried; techniques for reusing knowledge across time, domains, or tasks, by identifying similarities and adaptation to changes between them; techniques for making use of different types of information, such as labeled or unlabeled data, constraints or domain knowledge. Such techniques are studied for example in the fields of adaptive, active, semi-supervised, and transfer learning. However, this is mostly done in separate lines of research, while combinations thereof in interactive and adaptive machine learning systems that are capable of operating under various constraints, and thereby address the immanent real-world challenges of volume, velocity and variability of data and data mining systems, are rarely reported. Therefore, this workshop and tutorial aims to bring together researchers and practitioners from these different areas, and to stimulate research in interactive and adaptive machine learning systems as a whole. It continues a successful series of events at ECML PKDD 2017 in Skopje (Workshop & Tutorial), IJCNN 2018 in Rio (Tutorial), and ECML PKDD 2018 in Dublin (Workshop).

The workshop aims at discussing techniques and approaches for optimising the whole learning process, including the interaction with human supervisors, processing systems, and includes adaptive, active, semi-supervised, and transfer learning techniques, and combinations thereof in interactive and adaptive machine learning systems.

All in all, we accepted six regular papers (9 papers submitted) and three short papers (4 submitted) to be published in these workshop proceedings. The authors discuss approaches, identify challenges and gaps between active learning research and meaningful applications, as well as define new application-relevant research directions. We thank the authors for their submissions and the program committee for their hard work.

September 2019

Adrian Calma, Andreas Holzinger
Daniel Kottke, Georg Krempl, Vincent Lemaire

Organizing Committee

Adrian Calma, University of Kassel
Andreas Holzinger, Graz University of Technology
Daniel Kottke, University of Kassel
Georg Krempl, Utrecht University
Vincent Lemaire, Orange Labs France

Program Committee

Albert Bifet, LTCI, Telecom ParisTech
Giacomo Boracchi, Politecnico di Milano
Martin Holeňa, Institute of Computer Science, Prague
Edwin Lughofer, University of Linz
Christin Seifert, University of Twente
Jerzy Stefanowski, Poznan University of Poland
Indrė Žliobaitė, University of Helsinki

Table of Contents

Tutorial ... 1

Invited Talk ... 3

Full Research Papers ... 4

Short Papers ... 96

Foundations of Interactive Adaptive Learning

Georg Krempl

Information and Computing Sciences, Utrecht University, Utrecht
g.m.krempl@uu.nl

Abstract. This part starts with the classic stream mining paradigm. In its context, we discuss the challenges posed by non-stationarity and limitations in processing, storage, and supervision capacities. We briefly summarize related techniques, e.g. for incremental processing, forgetting, and change detection. Furthermore, we introduce techniques for optimising the interaction of a machine learning system with an oracle such as a human supervisor. We review active machine learning techniques, with focus on adaptive active learning for evolving and streaming data. We discuss recent advances and conclude with an overview on open research questions in adaptive active machine learning.

From Interactive Machine Learning to Explainable Artificial Intelligence (ex-AI)

Andreas Holzinger

Medical University of Graz, Graz
andreas.holzinger@medunigraz.at

Abstract. In this part, we focus on the role of humans in state-of-the-art decision systems. Thereby, we go beyond interactive machine learning to explainable artificial intelligence. How can this be realized? How can we include humans into the automated decision process and how can me measure their intelligence? To answer these questions, we will talk about different terms like interaction, reflection and discuss the underlying principles of intelligence and cognition. In the next part, we provide fundamentals to measure and evaluate human intelligence with biometric technologies, sensor arrays and affective computing to measure emotion and stress. The tutorial concludes with a discussion on ethical, legal and social issues of explainable AI systems.

Evaluation of Interactive Machine Learning Systems

Nadia Boukhelifa

INRA, Université Paris-Saclay, Paris
nadia.boukhelifa@inra.fr

Abstract. The evaluation of interactive machine learning systems remains a difficult task. These systems learn from and adapt to the human, but at the same time, the human receives feedback and adapts to the system. Getting a clear understanding of these subtle mechanisms of co-operation and co-adaptation is challenging. In this chapter, we report on our experience in designing and evaluating various interactive machine learning applications from different domains. We argue for coupling two types of validation: algorithm-centred analysis, to study the computational behaviour of the system; and human-centred evaluation, to observe the utility and effectiveness of the application for end-users. We use a visual analytics application for guided search, built using an interactive evolutionary approach, as an exemplar of our work. Our observation is that human-centred design and evaluation complement algorithmic analysis, and can play an important role in addressing the "black-box" effect of machine learning. Finally, we discuss research opportunities that require human-computer interaction methodologies, in order to support both the visible and hidden roles that humans play in interactive machine learning.

Toward Faithful Explanatory Active Learning with Self-explainable Neural Nets*

Stefano Teso

KU Leuven, Leuven, Belgium
stefano.teso@cs.kuleuven.be

Abstract. From the user's perspective, interaction in active learning is very *opaque*: the user only sees a sequence of instances to be labeled, and has no idea what the model believes or how it behaves. Explanatory active learning (XAL) tackles this issue by making the model predict and explain its own queries using local explainers. By witnessing the model's (lack of) progress, the user can decide whether to trust it. Despite their promise, existing implementations of XAL rely on post-hoc explainers, which can produce unfaithful and fragile explanations, which misrepresent the beliefs of the predictor, confuse the user, and affect the quality of her supervision. As a remedy, we replace post-hoc explainers with self-explainable models, and show how these can be actively learned from both labels and corrected explanations. Our preliminary results showcase the dangers of post-hoc explanations and hint at the promise of our solution.

Keywords: Machine Learning · Active Learning · Explainability

1 Introduction

Explainable machine learning has so far mostly focused on black-box models learned offline [8]. It was recently observed that interactive protocols can be black-box too [24, 16]. For instance, in active learning (AL), the user receives a sequence of instances (e.g. images, documents) to be labeled, but can witness neither the behavior of the predictor nor its beliefs.

Explanatory active learning (XAL) tackles this issue by injecting explanations into the learning loop [24]: whenever asking the user to label an instance, the model also shows a prediction for that instance and an explanation for the prediction. The explanations are obtained with a local explainer [14, 8], which summarizes and visualizes the local behavior of the predictor in terms of interpretable feature relevance or other understandable artifacts. By witnessing the evolution of the beliefs and decisions of the model, the user can justifiably grant

* The author is grateful to Kristian Kersting for many useful discussions and to Giuseppe Marra for help with the implementation. This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. [694980] SYNTH: Synthesising Inductive Data Models).

or revoke trust to it. This is analogous to trust between individuals, which requires to develop appropriate expectations through interaction [23]. In XAL, the user is also free to correct the explanations by, e.g., indicating any irrelevant or sensitive features that the model is currently relying on. This extra supervision is *necessary* to correct models that are “right for the wrong reasons” [19].

Existing implementations of XAL make use of *post-hoc* local explainers—for instance, CAIPI [24] uses LIME [18]—which treat the predictor being learned as a black-box. These approaches extract explanations through an approximate model translation [5] step. The overall process can produce unfaithful, fragile explanations that misrepresent the model’s beliefs and have high-variance [1, 2]. Unfaithful and unstable explanations may confuse the user and affect the quality of the user-provided corrections. More generally, such explanations are not trustworthy and conflict with the purpose of explanatory interaction.

As a remedy, we propose replacing post-hoc explainers with self-explainable neural networks (SENNs), a recently proposed class of models that automatically explain their own predictions. Intuitively, SENNs combine the transparency of linear models with the flexibility of neural nets [15]. The explanations produced by SENNs are exact, robust to small perturbations, and cheap to compute. In contrast with standard interpretable models (e.g. shallow decision trees), SENNs can tackle complex problems—including representation learning—via gradient descent. In order to integrate them into XAL, we show how to learn SENNs from labels and corrections directly by combining classification and ranking losses. Our preliminary empirical analysis shows that SENNs can substantially improve the quality of the explanations used in XAL and can be actively learned from labels and corrections.

Summarizing, we: 1) highlight the risks of unfaithful explanations in interactive learning; 2) propose a novel implementation of XAL based on self-explainable neural nets; 3) propose a joint loss to learn SENNs from labels and corrections directly; 4) report on preliminary experiments that showcase the behavior of post-hoc explainers and the promise of our solution.

2 Background

In the following, we will stick to binary classification and indicate instances as $x \in \mathcal{X}$ and labels as $y \in \mathcal{Y} = \{0, 1\}$. Our observations can be easily generalized to the multi-class case.

2.1 Post-hoc Local Explainers

Given a classifier $f : \mathcal{X} \to \mathcal{Y}$, for instance a neural network or a random forest, post-hoc local explainers explain individual predictions without looking at the exact inference steps performed by f [14]. Here we briefly detail LIME [18], which is central in current XAL implementations.

In order to explain a prediction $y^0 = f(x^0)$, LIME learns an *interpretable* local model g^0 that mimics f in the neighborhood of x^0, and then reads off an explanation from it. The local model is a sparse linear predictor or a shallow decision tree [18] built with (user-provided) interpretable features $\boldsymbol{\psi}(x)$. These may capture, e.g., individual words in document classification or objects in image tagging. The local model is learned from a synthetic dataset that describes counterfactual ("what if") information about switching on / off the interpretable features on $f(x^0)$.

More formally, the process amounts to:

1. Sampling s interpretable instances $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^s$ by randomly perturbing the interpretable representation of x^0, namely $\boldsymbol{\xi}^0 = \boldsymbol{\psi}(x^0)$;
2. Labeling each instance $\boldsymbol{\xi}^i$ using the target model $y^i = f(x^i)$, where $x^i = \boldsymbol{\psi}^{-1}(\boldsymbol{\xi}^i)$ is the pre-image of $\boldsymbol{\xi}^i$;
3. Weighting each example $(\boldsymbol{\xi}^i, y^i)$ by its similarity to $\boldsymbol{\xi}^0$, i.e., $k(\boldsymbol{\xi}^i, \boldsymbol{\xi}^0)$; the kernel function k determines the size and shape of the neighborhood of $\boldsymbol{\xi}^0$;
4. Fitting a local model g^0 on the synthetic dataset via *cost-sensitive learning*, so that examples outside of the neighborhood do not have much of an impact; this is a form of model translation [5];
5. Extracting an explanation from g^0. For instance, if g^0 is linear (i.e., $g^0(x) = \sum_j w_j \psi_j(x)$) then the explanation describes the contributions of the interpretable features according to the weights w_j. In practice only the largest weights are used. If g^0 is a decision tree, then the feature contributions can be read off from the path connecting the root to the predicted leaf.

The advantage of this procedure is that it is completely model-agnostic, as it treats f as a black-box. The downside, however, is that it is not exact, and so g^0 may not approximate f well. We will discuss the consequences later on.

2.2 Explanatory Active Learning

Pool-based active learning (AL) is designed for settings where labels are scarce and expensive to obtain [22, 9]. Learning proceeds iteratively. Initially, the model has access to a small set of labeled examples $\mathcal{L} \subseteq \mathcal{X} \times \mathcal{Y}$ and a large pool of unlabeled instances $\mathcal{U} \subseteq \mathcal{X}$. In each iteration, the model asks an oracle (i.e. a human expert or a measurement device) to label any instance in $\mathcal{U}$—*for a price*. The newly labeled instance is then moved to $\mathcal{L}$ and the model is adapted accordingly. These steps are repeated until a labeling budget is exhausted or the model is deemed good enough. The key challenge in AL is to design a strategy for selecting informative and representative query instances, so to learn good predictors at a small labeling cost. A common choice is uncertainty sampling (US) [13], which picks instances where the model is most uncertain in terms of, e.g., margin or entropy.

In order to make the interaction more transparent and directable, explanatory active learning (XAL) injects explanations into the learning loop and enables the user to interact with them [24]. In XAL, when the model asks the user to label an instance x, it also presents a prediction for that instance $\hat{y} = \hat{f}(x)$ and an explanation $\hat{z}$ for the prediction. In the CAIPI implementation of XAL, the explanation is computed with LIME. In exchange, the user provides the true label of x and optionally corrects the explanation. The correction indicates, for instance, which interpretable features (e.g. pixels, words) are erroneously being used by the model. Since models cannot learn from corrections directly, CAIPI converts corrections into *counter-examples*, as follows: if the user indicated that some interpretable feature $\boldsymbol{\psi}_i(x)$ is wrongly being used by the model, then CAIPI creates c copies of x where feature $\boldsymbol{\psi}_i$ is randomized, and attaches the true label to them. Intuitively, the counter-examples teach the predictor to predict the correct label independently from the value associated to the irrelevant feature. It was shown that, for LIME, explanation corrections describe local orthogonality constraints [24], and that counter-examples approximate these constraints.

By combining interactions and explanations, XAL helps the user to build a mental model of the predictor being learned, which is necessary for justifiably according trust to it. In addition, by virtue of learning from explanation corrections, XAL makes it less likely that the model learns to predict the right labels for the wrong reasons. We will see, however, that these promises can be difficult to keep when post-hoc explainers are involved.

2.3 Self-explainable Neural Networks

Sparse linear models over interpretable features are canonically considered to be highly interpretable. These models have the form[1] $f(x) = \sigma(\boldsymbol{w}^\top \boldsymbol{\phi}(x))$, where σ is a sigmoid function, $\boldsymbol{w} \in \mathbb{R}^n$ is constant, and $\boldsymbol{\phi} : \mathcal{X} \to \mathbb{R}^n$ is a fixed, interpretable feature map. The contribution of the jth feature to the output of the predictor is determined by the corresponding weight. Despite their interpretability, sparse linear models are limited to relatively simple learning tasks.

Self-explainable neural networks (SENNs) upgrade linear models by substantially increasing their capacity and flexibility while preserving their interpretability [15]. More specifically, SENNs have the same functional form, but allow the weights $\boldsymbol{w}$ to change across the space, i.e.:

$$f(x) = \sigma(\boldsymbol{w}(x)^\top \boldsymbol{\phi}(x)) \tag{1}$$

Here, both $\boldsymbol{w}(x)$ and $\boldsymbol{\phi}(x)$ are (arbitrarily deep) neural networks. The inner product can be replaced by any appropriate aggregation function [15]. In order to make $\boldsymbol{w}(x)$ act as an explanation, two additional restrictions are put in place:

1. The learned feature function $\boldsymbol{\phi}$ has to be interpretable. This can be achieved either by designing it by hand (as with linear predictors and LIME), by defining it in terms of (learned) prototypes, or by other means; see [15] for details.
2. As with linear models, the explanations should capture the local behavior of the model and not be affected by small displacements of the input instance. This is guaranteed by constraining $\boldsymbol{w}$ to vary slowly *with respect to* $\boldsymbol{\phi}$.

More formally, $\boldsymbol{w}$ is required to be locally difference bounded by $\boldsymbol{\phi}$, as per the following definition:

[1] Here and below, the bias term is left implicit.

Definition 1 ([15]). *A function* $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^a$ *is* locally difference bounded *by a function* $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^b$ *if for every* $\boldsymbol{x}^0 \in \mathbb{R}^n$ *there exists two constants* $\delta > 0$ *and* $L > 0$ *such that for every* $\boldsymbol{x} \in \mathbb{R}^n$*:*

$$\|\boldsymbol{x} - \boldsymbol{x}^0\| < \delta \implies \|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x}^0)\| \leq L\|\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{g}(\boldsymbol{x}^0)\|$$

In other words, for every point $\boldsymbol{x}^0$ there is a neighborhood within which[2] the change in $\boldsymbol{f}$ is bounded by the change in $\boldsymbol{g}$. Notice that the local Lipschitz "constant" L is allowed to vary with x^0.

This requirement is enforced during learning by penalizing the model for any deviations from linearity through the following regularization term:

$$\Omega(f) \stackrel{\text{def}}{=} \|\nabla_x f(x) - \boldsymbol{w}(x)^\top J_x\|$$

where J_x is the Jacobian matrix. The model f is learned by minimizing the empirical risk of some classification loss ℓ_Y plus the above regularizer, i.e.:

$$\min_f \ \hat{\mathbb{E}}_{(x,y)} \left[\ell_Y(f(x), y) + \alpha\Omega(f)\right] \tag{2}$$

Here $\hat{\mathbb{E}}$ indicates expectation over mini-batches and $\alpha \geq 0$ is a hyperparameter. As usual with neural networks, minimization is performed with gradient descent techniques. Of course, an additional term encouraging the output of $\boldsymbol{w}(x)$ to be sparse can be considered. We will see later on that this is automatically the case in our extension of SENNs to explanatory active learning.

It is worth pointing out that SENNs are quite different from attention models, which are yet another way of explaining (to some extent) neural networks. Indeed, the former explain exactly which features contribute to the prediction, as well as their polarity. The features themselves are arbitrary (interpretable) functions of the inputs. Attention models, in contrast, identify which *input features* (e.g. pixels) are relevant, but not how they contribute to the decision nor whether they are against/in favor.

3 Toward Faithful XAL

3.1 The Dangers of Post-hoc Explanations

Let us start by discussing the case of LIME, which is at the core of existing XAL implementations. In this case, the accuracy of the model translation process (described above) depends critically the choice of model class of g^0, interpretable features $\boldsymbol{\psi}$, kernel k, and number of samples s.

For instance:

1. If the kernel k is not chosen correctly (i.e. if it is too small, too broad, or has the wrong topology), then the synthetic dataset may fail to capture label changes around x^0, as shown in Figure 1.

[2] This can be limiting in classification scenarios, because the output should change abruptly when crossing the decision boundary. The formulation can be modified to account for this, but we keep it as is, for simplicity.

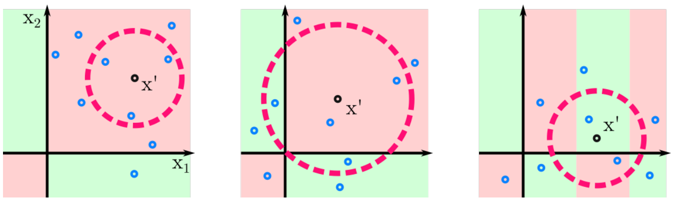


Fig. 1. Examples on which LIME produces unfaithful explanations. The decision surface of f is represented by the colored areas: light green means positive, light red negative. The pink circle represents the kernel k: points inside of it are assigned substantial weights while all others are not. Left: all synthetic examples with large weight have the same label. Middle: the synthetic examples fail to capture the non-additive interaction of the two features. Right: the kernel is too broad, and the synthetic dataset is highly complex and non-linear.

2. If the number of samples s is too small, the dataset may also not be representative.
3. $\boldsymbol{\psi}$ determines whether the pre-image of $\boldsymbol{z}^i$ is unique; if this is not the case, then the labeling step becomes unstable.

In all these cases, LIME produces high-variance explanations that substantially misrepresent the behavior of f. Increasing the number of samples may improve the situation, but also the runtime, and may not be enough to stabilize the explanations anyway.

Crucially, the same high-level argument applies to all post-hoc explainers, because all of them treat f as a black-box and thus—by construction—must include an inexact model translation step.

From the standpoint of XAL, unfaithfulness has two major consequences. First, high-variance explanations portray the model as behaving semi-randomly, regardless of whether it is good or not. The user may also perceive that her supervision has no effect, and feel lack of control. In rare cases, CAIPI may also accidentally persuade the user into trusting a bad model. Both cases are problematic in sensitive applications, like the ones that XAL is designed for. More generally, unfaithful explanations misrepresent the learned model, defeating the purpose of explanatory active learning. Second, unfaithful explanations compromise the usefulness of the corrections. If the user is confused by the explanations, her corrections will be not as informative. Further, the correction may specify not to use a feature that the model is not using anyway (or vice versa). Depending on the model being learned, correcting the same feature too many times may also lead to learning instabilities.

It is therefore desirable to fix the XAL pipeline to rely on faithful and trustable explanations. In the next section, we show how to do so.

Algorithm 1 Pseudocode of CALI: $\mathcal{L}$ is the set of labeled examples, $\mathcal{U}$ is the set of unlabeled instances, and T is the query budget.

```
 1: procedure CALI($\mathcal{L}$, $\mathcal{U}$, $T$)
 2:     $\mathcal{C} \leftarrow \varnothing$
 3:     $f \leftarrow \mathrm{fit}(\mathcal{L}, \mathcal{C})$                                    ▷ Eq. 3
 4:     repeat
 5:         $x \leftarrow \mathrm{select}(f, \mathcal{U})$,  $\hat{y} \leftarrow f(x)$,  $\hat{\boldsymbol{z}} \leftarrow \boldsymbol{w}(x)$
 6:         Present instance $x$, prediction $\hat{y}$, and explanation $\hat{\boldsymbol{z}}$ to the user
 7:         Receive label $y$ and explanation correction $\bar{\boldsymbol{z}}$
 8:         Remove $x$ from $\mathcal{U}$, add $(x, y)$ to $\mathcal{L}$, add $\bar{\boldsymbol{z}} \succeq \hat{\boldsymbol{z}}$ to $\mathcal{C}$
 9:         $f \leftarrow \mathrm{fit}(\mathcal{L}, \mathcal{C})$                                ▷ Eq. 3
10:     until budget $T$ is exhausted or $f$ is good enough
11:     return $f$
```

3.2 Explanatory Active Learning with SENNs

Our proposed algorithm—dubbed Calimocho, or CALI for short—follows closely the XAL learning loop; the pseudocode is listed in Algrithm 1. Notice that f here is a SENN. Explanation corrections are collected in a set $\mathcal{C}$, initially empty. In each iteration, the algorithm chooses an instance $x \in \mathcal{U}$ (using uncertainty sampling, as CAIPI, for simplicity), predicts its class $\hat{y}$, and generates an explanation $\hat{\boldsymbol{z}}$ for the prediction (line 5). The explanation $\hat{\boldsymbol{z}}$ is simply read off of $\boldsymbol{w}(x)$, i.e., $\hat{\boldsymbol{z}} = \boldsymbol{w}(x)$, without any projection or sampling step. All components are presented to the user (lines 6–7), who replies with the true label y of x and optionally with an improved explanation $\bar{\boldsymbol{z}}$. Finally, the dataset is updated and the preference $\bar{\boldsymbol{z}} \succeq \hat{\boldsymbol{z}}$ is added to the corrections $\mathcal{C}$.

The learning step requires to strategy to train SENNs using corrections too. One option is to follow CAIPI, an use counter-examples. However, these only approximately capture the constraint imposed by the correction, e.g., that the label should not depend on a particular interpretable feature. Depending on the application, there is also a (slim) chance that the counter-examples are actually wrong. Feature influence supervision solves this issue by asking the user [20], but this can be cognitively costly.

Instead, we opt for learning SENNs directly from corrections, as follows. Recall that, given an explanation $\hat{\boldsymbol{z}}$, a correction specifies, e.g., which features are erroneously being used by the model. Applying the correction to $\hat{\boldsymbol{z}}$ leads to a corrected explanation $\bar{\boldsymbol{z}}$. It is therefore natural to impose that $\boldsymbol{w}(x)$ generates explanations that are closer to the corrected explanation rather than the predicted on. This can be accomplished by, e.g., minimizing the squared Euclidean distance $\|\boldsymbol{w}(x) - \bar{\boldsymbol{z}}\|_2^2$ while maximizing $\|\boldsymbol{w}(x) - \hat{\boldsymbol{z}}\|_2^2$. It is easy to see that this is equivalent to imposing a ranking loss:

$$\|\boldsymbol{w}(x) - \bar{\boldsymbol{z}}\|_2^2 - \|\boldsymbol{w}(x) - \hat{\boldsymbol{z}}\|_2^2 = 2\langle \boldsymbol{w}(x), \hat{\boldsymbol{z}} - \bar{\boldsymbol{z}} \rangle + \left(\|\bar{\boldsymbol{z}}\|_2^2 - \|\hat{\boldsymbol{z}}\|_2^2 \right)$$

Notice that the features that were *not* corrected by the user (i.e. $\bar{z}_j - \hat{z}_j = 0$) do not contribute to the loss, as expected. Our solution also generalizes the input gradient constraints introduced in [19] from learning from full explanations to corrections, which contain information about only few interpretable features.

Letting the dataset include instances x, labels y, and preferences $\bar{\boldsymbol{z}} \succeq \hat{\boldsymbol{z}}$ (represented with $\mathcal{C}$ in the pseudo-code), and denoting the ranking loss as $\ell_Z(\boldsymbol{w}(x), \bar{\boldsymbol{z}} \succeq \hat{\boldsymbol{z}}) = \langle \boldsymbol{w}(x), \hat{\boldsymbol{z}} - \bar{\boldsymbol{z}} \rangle$, CALI fits f by extending Eq. 2 with the above loss:

$$\min_f \ \hat{\mathbb{E}}_{(x, \bar{\boldsymbol{z}} \succeq \hat{\boldsymbol{z}}, y)} \left[\lambda \ell_Y(f(x), y) + (1 - \lambda) \ell_Z(\boldsymbol{w}(x), \bar{\boldsymbol{z}} \succeq \hat{\boldsymbol{z}}) + \alpha \Omega(f) \right] \qquad (3)$$

Noisy corrections can be handled by tweaking the hyperparameter $\lambda \in [0, 1]$.

3.3 Discussion

A few remarks are in order. First, CALI trades off the model-agnosticism of CAIPI for exactness and efficiency. This is desirable, since faithfulness is necessary to justifiably establish trust, especially in sensitive applications. In addition, despite their apparent simplicity, SENNs are very flexible non-linear models [15], substantially more powerful than "standard" interpretable models (e.g. they natively support representation learning) and easy to learn with state-of-the-art gradient descent techniques.

It is natural to ask what kind of SENN architectures can be learned in a label-scarce framework like active learning. On the one hand, shallow SENNs can be learned efficiently from few labels alone. On the other, by supplementing label information, we expect explanation corrections do help to actively learn deep(er) models. Our preliminary results suggest that this might be the case.

4 Experiments

We address the following research questions:

Q1 Are the explanations output by LIME faithful?
Q2 Does CALI learn from corrections?
Q3 Does explanatory feedback help learn deeper models?

In order to do so, we implemented CALI[3] and ran a preliminary experiment with the synthetic color dataset used in [19, 24], where the goal is to classify small synthetic images *for the right reasons.* The images are 5 × 5 and have four possible colors. An image is positive if i) either the four corners have the same color, or ii) the top three middle pixels all have different colors. On all training images either both rules are satisfied or neither is. This means that labels alone are not enough to disambiguate between the two potential explanations. Both images and explanations are represented as $5 \times 5 \times 4$ one-hot arrays. Notice that the two conditions can be easily expressed as linear concepts in this space. As in [24], we consider true explanations $\boldsymbol{z}$ that highlight the k most relevant pixels. The corrections instead highlight (a subset of the) pixels that are wrongly

[3] Code at: `github.com/stefanoteso/calimocho`

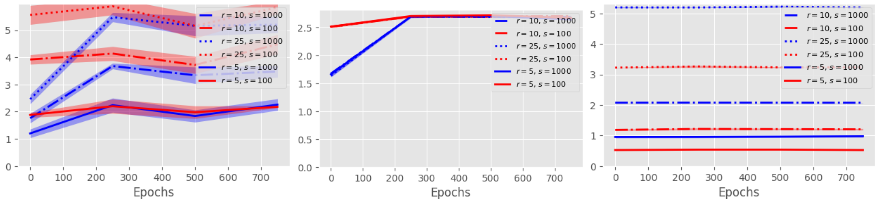


Fig. 2. Quality of the explanations produced by LIME for shallow SENNs trained passively for 1000 epochs (x axis); s is the number of samples used by LIME, while r is the number of repeats. Left: similarity between LIME explanations and the true explanations in terms of recall@k. Middle: diversity between LIME explanations across re-runs in terms of average pair-wise Euclidean distance between the r explanations. Right: runtime of LIME in seconds.

highlighted in the predicted explanations $\hat{\boldsymbol{z}}$. In all cases, we use a SENN where $\boldsymbol{\phi}(x)$ is simply the one-hot encoding of x and $\boldsymbol{w}(x)$ is a fully-connected feed-forward neural network with $L = 1, 3, 5$ hidden layers. All results are 5-fold cross-validated.

Q1: Are the explanations output by CAIPI *faithful?* We trained a SENN for 1000 epochs and every 250 iterations computed the recall@k of the explanations produced by LIME on 10 random (but fixed) test examples. This was done by looking at how many of the k highest scoring features in the SENN explanation (which is exact) appear among the k highest scoring features found by LIME. In practice, CAIPI stabilizes high-variance explanation by running LIME r times and averaging the obtained explanations. To check whether this technique is effective, we repeated LIME $r = 5, 10, 25$ times and then measured the pair-wise Euclidean distance between the r explanations. The results, in Figure 2, show that the LIME explanations are never completely faithful, regardless of the number of samples s and repeats r, thus confirming our arguments about post-hoc explainers. In addition, while increasing s does have a clear beneficial effect, r is surprisingly not beneficial. Regardless, increasing either does have an effect on runtimes, as shown by the right plot. In comparison, SENN explanations are exact and robust by construction and also essentially free to compute, because $\boldsymbol{w}(x)$ must be evaluated *anyway* whenever doing inference. In practice, evaluating $\boldsymbol{w}$ amounts to forward propagation and takes a tiny fraction of a second (data not shown). This can be a substantial advantage in interactive applications to avoid making the user wait.

Q2: Does CALI *learn from corrections?* In this experiment, we select each rule in turn and use CALI to actively learn a SENN from explanation corrections. The ground-truth explanation highlight the 4 or 3 pixels that the decision actually depends on, and the corrections identify the $c = 1, \dots, 4$ pixels whose predicted weight $w_i(x)$ is farthest away from the correct weight. Figure 3 shows the label loss ℓ_Y and the explanation loss ℓ_Z as more queries are made, up to 300. Turning

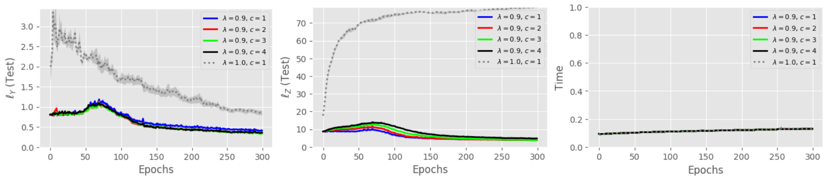


Fig. 3. Test set performance of CALI when learning from explanation corrections as more queries are asked (x axis). Left: label loss ℓ_Y. Middle: explanation loss ℓ_Z. Right: per-iteration runtime in seconds.

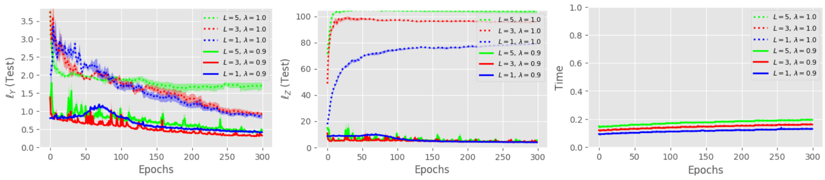


Fig. 4. Test set performance of CALI on progressively deeper SENNs. as more queries are asked (x axis). Left: label loss ℓ_Y. Middle: explanation loss ℓ_Z. Right: per-iteration runtime in seconds.

on learning from corrections brings a huge benefit, as expected. Indeed, when no corrections are provied (i.e. $\lambda = 1$), the SENN slowly learns the target concept, as shown by the leftmost plot, while corrections help the SENN to converge much faster. Even more importantly, unless corrections are enabled, the model fails to be "right for the right reasons", as the explanation loss *diverges* as more labels are obtained (middle plot). Finally, the rightmost plot shows that active learning CALI is very efficient, requiring less than 0.2 seconds per iteration on average. One surprising result is that increasing the number of corrections c does not monotonically increase performance. This needs to be validated further; however, the overall trend is clear and is consistent with the findings of [24]. The results for the two rules and different query strategies (random and margin-based uncertainty sampling) show exactly the same trend, and are not reported.

Q3: Does explanatory feedback help learn deeper models? Finally, we look at the effect of learning from corrections while increasing the number of hidden layers $L = 1, 3, 5$ of $\boldsymbol{w}(x)$. The results of increasing L are reported in Figure 4. Once again, the effect of corrections is very clear: they help the label loss to decay faster, and are necessary for the explanation loss not to diverge. Most importantly, these results hold regardless of the choice of L, with larger models behaving much worse on explanation loss when learned from labels only. This result is (preliminary but) very interesting especially in the light of the observation that CAIPI behaves best when learning sparser models [24]. In stark contrast, CALI seems to behave well even for deeper SENNs.

5 Related Work

Despite the surge of interest on explainable AI and machine learning, most research has focused on passive learning, and specifically on (1) designing interpretable predictors [25, 12] and (2) explaining black-box models such as neural networks [3, 8]. Instead, we consider explanations in an interactive learning setting.

We specifically study explanatory active learning (introduced in [24] in the wider context of explanatory *interactive* learning) which injects explanations into the active learning loop. Related approaches were proposed in [16], which uses LIME to communicate the exploration pattern of the active learner to the user, and in [20], where a model is learned explicitly from feature-level feedback. Like previous work on (interactive) feature selection and dual supervision [17, 7, 4, 21], these work either ignore the issue of trust or the advantages of learning from corrections rather than explanations. Please see [24] for a discussion. Conceptually, XAL is related to techniques like explanatory debugging [11], which however targets simple predictors only. Similar themes have been championed by [10].

The importance of explanation faithfulness has long been recognized [3]. More recently, the hidden dangers of local explainability tools have been the subject of a number of studies, e.g., [26] and [1], and the limitations of post-hoc explainers have been studied in [2]. The issue of unfaithful explanations in explanatory interactive learning, however, has not been considered before. These studies have lead to developing exact and / or robust explanatory techniques like input gradients [19] (aka instantaneous causal effects), average causal effects [6], and of course SENNs [15]. To the best of our knowledge, however, SENNs are the only method that 1) supports gradient-based optimization and interpretable representation learning, 2) generates exact explanations that are robust to perturbations, and 3) can be learned from labels and—with this paper—from corrections directly.

6 Conclusion and Outlook

The present paper highlights the dangers of post-hoc explainers for explanatory active learning (XAL). Post-hoc explainers are prone to generating unfaithful explanations that misrepresent the target predictor and prevent the user from appropriately allocating trust to it. In order to solve this issue, we extend existing XAL implementations by replacing post-hoc explainers with self-explainable neural networks (SENNs). SENNs automatically generate exact and robust explanations for their own predictions. In order to integrate them with XAL, we show how to learn SENNs from labels and explanation corrections by combining classification and ranking losses. Our preliminary experiments showcase the fragility of post-hoc explainers and the potential of SENNs in explanatory active learning.

Of course, our results need further validation, including a more direct comparisong with CAIPI, which we plan to carry on soon. They also hint at the promise of corrections for learning deeper networks even when labels are scarce. We plan to investigate this direction in future work.

References

1. Adebayo, J., Gilmer, J., Goodfellow, I., Kim, B.: Local explanation methods for deep neural networks lack sensitivity to parameter values. arXiv e-prints (Oct 2018), arXiv:1810.03307
2. Alvarez-Melis, D., Jaakkola, T.S.: On the robustness of interpretability methods. arXiv e-prints (Jun 2018), arXiv:1806.08049
3. Andrews, R., Diederich, J., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-based systems **8**(6), 373–389 (1995)
4. Attenberg, J., et al.: A unified approach to active dual supervision for labeling features and examples. Machine Learning and Knowledge Discovery in Databases pp. 40–55 (2010)
5. Buciluǎ, C., et al.: Model compression. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 535–541. ACM (2006)
6. Chattopadhyay, A., Manupriya, P., Sarkar, A., Balasubramanian, V.N.: Neural network attributions: A causal perspective. In: International Conference on Machine Learning. pp. 981–990 (2019)
7. Druck, G., et al.: Active learning by labeling features. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1. pp. 81–90. Association for Computational Linguistics (2009)
8. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM computing surveys (CSUR) **51**(5), 93 (2018)
9. Hanneke, S.: Theory of disagreement-based active learning. Foundations and Trends® in Machine Learning **7**(2-3), 131–309 (2014)
10. Holzinger, A.: From machine learning to explainable ai. In: 2018 World Symposium on Digital Intelligence for Systems and Machines (DISA). pp. 55–66. IEEE (2018)
11. Kulesza, T., et al.: Principles of explanatory debugging to personalize interactive machine learning. In: Proceedings of the 20th International Conference on Intelligent User Interfaces. pp. 126–137. ACM (2015)
12. Lage, I., Ross, A., Gershman, S.J., Kim, B., Doshi-Velez, F.: Human-in-the-loop interpretability prior. In: Advances in Neural Information Processing Systems. pp. 10159–10168 (2018)
13. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: SIGIR'94. pp. 3–12. Springer (1994)
14. Lundberg, S., et al.: An unexpected unity among methods for interpreting model predictions. arXiv e-prints (Nov 2016), arXiv:1611.07478
15. Melis, D.A., Jaakkola, T.: Towards robust interpretability with self-explaining neural networks. In: Advances in Neural Information Processing Systems. pp. 7775–7784 (2018)
16. Phillips, R., Chang, K.H., Friedler, S.A.: Interpretable active learning. In: Conference on Fairness, Accountability, and Transparency (2018)
17. Raghavan, H., et al.: Active learning with feedback on features and instances. Journal of Machine Learning Research **7**(Aug), 1655–1686 (2006)

18. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should I trust you?: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1135–1144. ACM (2016)
19. Ross, A.S., Hughes, M.C., Doshi-Velez, F.: Right for the right reasons: training differentiable models by constraining their explanations. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp. 2662–2670. AAAI Press (2017)
20. Sen, S., Mardziel, P., Datta, A., Fredrikson, M.: Supervising feature influence. arXiv e-prints (Mar 2018), arXiv:1803.10815
21. Settles, B.: Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In: Proceedings of the conference on empirical methods in natural language processing. pp. 1467–1478. Association for Computational Linguistics (2011)
22. Settles, B.: Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning **6**(1), 1–114 (2012)
23. Simpson, J.A.: Psychological foundations of trust. Current directions in psychological science **16**(5), 264–268 (2007)
24. Teso, S., Kersting, K.: Explanatory interactive machine learning. In: Proceedings of AIES'19 (2019)
25. Wu, M., Hughes, M.C., Parbhoo, S., Zazzi, M., Roth, V., Doshi-Velez, F.: Beyond sparsity: Tree regularization of deep models for interpretability. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
26. Yeh, C.K., Hsieh, C.Y., Sai Suggala, A., Inouye, D., Ravikumar, P.: On the (In)fidelity and Sensitivity for Explanations. arXiv e-prints (Jan 2019), arXiv:1901.09392

Validating One-Class Active Learning with User Studies – a Prototype and Open Challenges

Holger Trittenbach, Adrian Englhardt, and Klemens Böhm

Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
{holger.trittenbach, adrian.englhardt, klemens.boehm}@kit.edu

Abstract. Active learning with one-class classifiers involves users in the detection of outliers. The evaluation of one-class active learning typically relies on user feedback that is simulated, based on benchmark data. This is because validations with real users are elaborate. They require the design and implementation of an interactive learning system. But without such a validation, it is unclear whether the value proposition of active learning does materialize when it comes to an actual detection of outliers. User studies are necessary to find out when users can indeed provide feedback. In this article, we describe important characteristics and prerequisites of one-class active learning for outlier detection, and how they influence the design of interactive systems. We propose a reference architecture of a one-class active learning system. We then describe design alternatives regarding such a system and discuss conceptual and technical challenges. We conclude with a roadmap towards validating one-class active learning with user studies.

Keywords: Active learning · One-class classification · Outlier detection · User study.

1 Introduction

Active Learning is the paradigm to involve users in classifier training, to improve classification results by means of feedback. An important application of active learning is outlier detection. Here, one-class classifiers learn to discern inliers from outliers. Examples are network security [17,37] or fault monitoring [45]. In these cases, one-class classifiers with active learning (OCAL) ask users to provide a binary label ("inlier" or "outlier") for some of the observations. Then they use these labels in subsequent training iterations to learn an accurate decision boundary. OCAL differs from other active learning applications such as balanced binary or multi-class classification. This is because the strategies to select observations for feedback ("query strategies") take into account that outliers are rare to non existent.

Over the last years, several OCAL-specific query strategies have been proposed. They focus on improving classification accuracy with minimal feedback [3, 15, 16, 18, 45]. To evaluate them experimentally, authors generally rely on data sets with an available ground truth, in order to simulate user feedback. On the one hand, this seems to be convenient, since it allows to evaluate algorithmic improvements without a user study. On the other hand, simulating user feedback requires some assumptions on how users give feedback. These assumptions are generally implicit, and authors do not elaborated on them, since they have become widely accepted in the research community. In particular, there are two fundamental *assumptions behind active learning*:

(Feedback) Users provide accurate feedback independently from the presentation of the classification result and from the observation selected for feedback (the “query”).

(Acceptance) Users do not question how the feedback provided changes the classification results. Their motivation to provide feedback, even for many observations, does not hinge on their understanding of how their feedback influences the classifier.

Think of an OCAL system that asks a user to provide a class label on a 20-dimensional real-valued vector, where features are the result of some preprocessing, such as principal component analysis. We argue that, without further information, users are unlikely to comprehend the query, and cannot provide accurate feedback. This is in line with observations from literature [14,33]. Even if they could provide the feedback, any change in the classifier is not tangible. This is because there is no suitable visualization or description of a 20-dim decision boundary. We argue that users may question whether their feedback has a positive effect on the classifier, or even any effect at all, lose interest, and eventually discontinue to provide feedback.

This is a peculiar situation: On the one hand, the value proposition of active learning is to obtain helpful information from users that is not yet contained in the training data. On the other hand, there currently is no validation whether one can realize this value in an actual application. Such a validation would require to implement an interactive OCAL system and to conduct a user study.

However, such an experimental validation is difficult, since there are several conceptual and technical issues. We have experienced this first hand, when we have looked at smart meter data of an industrial production side [7,42] to identify unusual observations, and to collect respective ground truth labels from human experts. In our preliminary experiments with this data, we found that both *active-learning assumptions do not hold* in practice. In particular, we have observed that domain experts ask for additional information such as visualizations and explanations that go way beyond a simple presentation of classification results. Since there is an over-reliance on active-learning assumptions, only little effort has been spent on making OCAL interpretable, comprehensible, and usable. So it is unclear what the minimal requirements behind an OCAL system are to carry out a user study. Second, there are *conceptual issues* that are in the way of implementing OCAL systems. One issue is that the design space of OCAL systems is huge. It requires to define a learning scenario, to choose a suitable classifier and a learning strategy, as well as selecting multiple hyperparameter values. In addition, there may be several conflicting objectives: One may strive to improve classification accuracy. Another objective may be to use OCAL

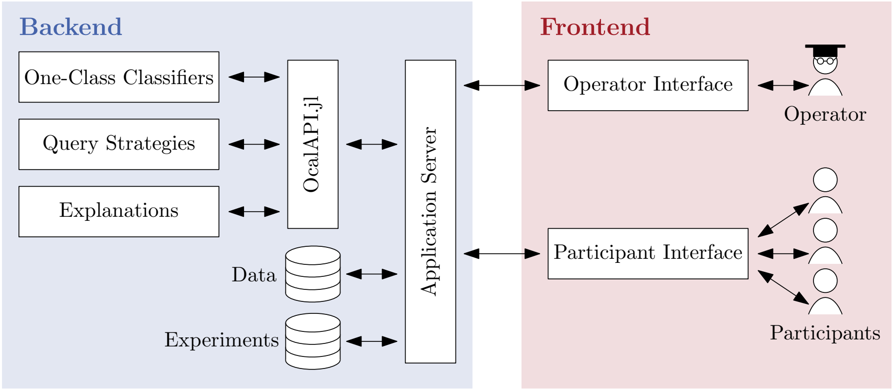


Fig. 1. System overview

as an exploratory tool, to present users with as many interesting instances as possible. Another issue is that objectives of a user study are diverse. One may want to collect a reliable ground truth for a novel data set, or to evaluate specific components of the active learning system, e.g., how well users respond to a particular visualization. Third, there are *technical issues* which we have experienced first hand when implementing an OCAL system prototype. For instance, runtimes of training state-of-the-art classifiers may be too long for interactivity. Another example is that it is unclear how to visualize decision boundaries in multi-dimensional data sets.

Although there are many difficulties, we deem user studies imperative to understand the determining factors behind realizing the value of OCAL. These factors can serve as guidelines for data mining research and can eventually lead to a more differentiated evaluation of novel query strategies and classifiers. The objective of our article is to point out important characteristics and prerequisites of OCAL and how they influence the design of interactive systems. To our knowledge, this is the first overview on conceptual and technical challenges regarding OCAL systems. We derive these challenges based on an architectural sketch on the components of an existing OCAL system, which we have implemented as a prototype. We conclude by proposing a roadmap towards validating OCAL with user studies.

2 OCAL System Architecture

The purpose of an OCAL system is to facilitate experiments with several users. An experiment is a specific technical configuration, i.e., a data set, a classifier, a query strategy, and one or more users, the participants, who provide feedback.

An OCAL system consists of several modules. Participants interact with the system through a *participant interface* that visualizes information on active learning iterations, such as the classification result and the progress of the experiment. The training of the classifier, query selection, and the preparation of additional information such as visualizations and explanations take place in an *algorithm backend*. Finally, there is a human operator who configures, monitors and evaluates the experiments through an *operator interface*. This typically is the researcher who conducts the experiments. Figure 1 is an overview of the system architecture. In the following, we describe the different modules and link them to our prototype implementation.

Algorithm Backend: On a technical level, the algorithm backend consists of a classifier module *SVDD.jl*[1] and a module *OneClassActiveLearning.jl*[2], which implements active learning components such as the query strategies. A third module provides additional information, e.g., classifier visualizations. For our prototype, we have implemented the classifiers, query strategies and basic visualization information in *OcalAPI.jl*[3], a ready-to-use JSON REST API. This decoupling allows to re-use the algorithm backend independent of the participant and operator interface.

Operator Interface: The operator interface allows an operator to configure so-called experiment *setups*. A setup consists of a data set, a parameterized classifier and a query strategy. Depending on the research question, the operator may also configure which information is displayed in the participant interface. This gives way to A/B tests, to, say, validate if a certain visualization has an effect on feedback quality. The operator can invite several users to participate in an *experiment run*, i.e., an instantiation of an experiment setup. He can monitor and inspect the experiment runs in an overview panel and export experiment data for further analysis.

Participant Interface: The participant interface has two functions. First, it is an input device to collect feedback during the experiment. Second, it provides the participants with information that supports them to provide educated feedback. For instance, this may be a visualization of a classifier, a view on the raw data or a history of classification accuracy over the past iterations. The participant then provides feedback for some observations. During this process, the interface captures user interactions, e.g., mouse movement and selection. When the query budget or time limit is not exhausted, the participant proceeds with the next iteration.

Our implementation of the interfaces is preliminary, since there are several open challenges, both conceptual and technical (see Section 3). We plan to make

[1] `https://github.com/englhardt/SVDD.jl`
[2] `https://github.com/englhardt/OneClassActiveLearning.jl`
[3] `https://github.com/englhardt/OcalAPI.jl`

it publicly available in the future as well. An important takeaway from this section is an intuition about how OCAL systems can be designed, on an architectural level. This intuition may be useful to understand the following discussions on the design space of OCAL systems and the challenges related to the three modules.

3 Design Decisions for OCAL Systems

The design and implementation of OCAL systems are inherently interdisciplinary and require expertise from several areas, including data mining, human-computer interaction, UX-design, and knowledge of the application domain. Although all disciplines are important, we now focus on the data mining perspective. We first discuss different types of interaction and elaborate on the design options for one-class classifiers and query strategies. We then present different options to prepare information for users during the learning iterations. Finally, we elaborate on several technical challenges when implementing OCAL systems.

3.1 Type of Interaction

The common definition of active learning is that a query strategy selects one or more observations for feedback. So, strictly speaking, a user does not have the option to also give feedback on other observations not selected by the system. However, there are related disciplines that do away with this restriction. For instance, one research direction is Visual Interactive Analytics (VIA) [8,25,43], where a user interactively explores outliers in a data set. VIA systems provide different kinds of visualization to assist users in identifying outliers, in particular with high-dimensional data sets. The unification of active learning and VIA is Visual Inter-Active Labeling (VIAL) [6,27]. VIAL combines active learning with user-supporting visualizations from the VIA community. Variants of VIAL and active learning are conceivable as well. For instance, instead of asking for labels of specific observations, the query strategy could provide a set of observations from which users can select one or more to label.

It is an open question in which cases one should use VIAL or active learning. A user study in [5] indicates that users label more observations if they are free to choose the observations. However, the resulting classifier accuracy is higher with an AL query strategy. It is unclear whether these insights transfer to outlier detection where classes are unbalanced. In fact, we see this as one of the overarching questions to answer with user studies.

3.2 Type of Feedback

In this article, feedback is binary, i.e., users decide whether an observation belongs to the inlier or outlier class. However, other types of feedback are conceivable as well. For instance, in multi-class settings, the system may ask users to state to which classes an observation does *not* belong [10]. Another example is to ask users for feedback on features, as opposed to instances [13]. Existing OCAL approaches in turn focus on binary feedback. It is an open question if and how OCAL can benefit from allowing for different types of feedback.

3.3 OCAL Design Space

An OCAL system consists of three building blocks: the learning scenario, the classifier, and the query strategy. In brief, a *learning scenario* are underlying assumptions about the application and user interaction. This includes the feedback type, e.g., sequential labels, the budget available for feedback, as well as assumptions on the data distribution, the objective of the learning process, e.g., to improve the accuracy of the classifier, and an initial setup, which includes how many labels are available when the active learning starts. In addition, there are several semi-supervised classifiers, such as SVDDneg [39], and *query strategies*, e.g., high-confidence sampling [3], which one can combine almost arbitrarily with any of the learning scenarios. Almost all classifiers and query strategies require to set additional hyperparameters. Their value can have significant influence on result quality, and a poor choice may bog it down. Moreover, a good query strategy and hyperparemeter values may also depend on the active learning progress, i.e., the number of labels already provided by the user.

Navigating this design space is challenging, and it is generally not feasible to consider and evaluate all possible combinations. Although there is an overview and a benchmark on OCAL [41], a good solution still is application-specific and may require fine-tuning of several components.

3.4 Preparation of Information

Classifier training and query selection produce a lot of data. On a fine-granular level, this includes the parameterized decision function for the classifier and informativeness scores for the query strategy. After processing this data, query strategies select the most informative instances and predict a label for each observation. In general, this data can be processed and enriched in many ways before presenting it to a user. On a coarse level, one can provide users with additional information, such as explanations of the classifier or contextual information on the learning progress. We now discuss several types of information to present during an active learning iteration: the query, the result, black-box explanations and contextual information.

Query presentation: After selecting observations for feedback, “queries” in short, they must be presented to the user. In general, there are two representations of a query. First, the query has a *raw-data representation*. Examples are text documents, multimedia files, multi-dimensional time series of real-valued sensors, or sub-graphs of a network. Second, the data often is pre-processed to a *feature representation*, a real-valued vector that the classifier can process. In principle, queries can be presented to users in either representation. Our experience is that domain experts are more familiar with raw data and demand it even if the feature representation is interpretable.

Next, one can provide context information for queries. For an individual instance, one can show the nearest neighbors of the query or a difference to prototypes of both classes. Another approach is to use visualization techniques for high-dimensional data [28, 32] to highlight the query. One can also visualize the score distribution over all candidate queries. Depending on the type of the query strategy, it also is possible to generate heatmaps that indicate areas in the data space with high informativeness [44] together with the query.

Result presentation: The presentation of a classification result largely depends on the classifier. With OCAL, the classifiers predominantly used rely on the notion of Support Vector Data Description (SVDD) [39]. In a nutshell, SVDD is an optimization problem to fit a hypersphere around the data, while allowing a small percentage of the data, the outliers, to lie outside the hypersphere. By using the kernel trick, the decision boundaries can become of arbitrary shape. So a natural presentation of SVDD is a contour plot that shows distances to the decision boundary. However, when data has more than two dimensions, contour plots are not straightforward. The reason is that contour plots rely on the distance to the decision boundary for a two-dimensional grid of observations (x_1, x_2). However, the distance depends on the full vector $(x_1, x_2, \ldots, x_n)$ and thus cannot be computed for low-dimensional projections. One remedy would be to train a classifier for each of the projections to visualize. However, the classifier trained on the projection may differ significantly from the classifier trained on all dimensions. So a two-dimensional contour plot may have very little benefit. With common implementations of one-class classifiers, one is currently restricted to present results as plain numeric values, raw data, and predicted labels.

Black-Box Explanations: Orthogonal to inspecting the queries and the classification result, there are several approaches to provide additional explanations of the classification result. The idea is to treat the classifier, or more generally any predictive model, as a black box, and generate post-hoc explanations for the prediction of individual observations. This is also called *local explanation*, since explanations differ between instances. Recently, CAIPI, a local explainer based on the popular explanation framework LIME [30], has been proposed to explain classification results in an active learning setting [40]. The idea behind CAIPI is to provide the user with explanations for the prediction of a query and ask them to correct wrong explanations. Another application of LIME is to explain why an observation has been selected as a query [29]. The idea behind this approach is to explain the informativeness of a query by its neighborhood. The authors use uncertainty sampling, and this approach may also work with other query strategies, such as high-confidence sampling [3]. However, with more complex query strategies, for instance ones that incorporate local neighborhoods [45] or probability densities [16], applying LIME may not be straightforward. For outlier detection, there exist further, more specific approaches to generate explanations for outlierness. An example is to visualize two-dimensional projections for input features that contribute most to an outlier score [19]. Other examples are methods from the VIA community that allow users to explore outliers interactively [8, 25, 43].

Contextual Information: The participant interface can also provide additional information that spans several active learning iterations. For instance, the interface can give users access to the classification history, allow them to revisit their previous responses, and give them access to responses of other users, if available. This can entail several issues, such as how to combine possibly diverging responses from different users, and the question whether users will be biased by giving them access to feedback of others. Studying such issues is focus of collaborative interactive learning [9]. Others have proposed to give users access to 2D scatter plots of the data, the confusion matrix and the progress of classification accuracy on labeled data [24]. In this case, accuracy measures may be biased. For instance, after collecting a ground truth for the first few labels, accuracy may be very high. It may decrease when more labels become available, and the labeled sample covers a larger share of the data space. So it remains an open question whether contextual information will indeed support users to provide accurate feedback.

To conclude, one faces many options in the design of OCAL systems. In particular, there are many approaches to support users with information so that they can make informed decisions on the class label. However, the approaches discussed have not yet been evaluated by means of user studies. Instead, they are limited to a theoretical discussion, simulated feedback based on benchmark data, or pen and paper surveys [40]. It is largely unclear which methods do enable users to provide feedback and indeed improve the feedback collected.

3.5 Technical Challenges

Active learning induces several technical requirements to make systems interactive, and to collect user feedback. Most requirements are general for active learning systems But their realization with one-class classifiers is difficult.

Cold Start In most cases, active learning starts with a fully unsupervised setting, i.e., there is no labeled data available. This restricts the possible combinations of classifiers and query strategies in two cases. First, some query strategies, e.g., sampling close to the decision boundary, require a trained one-class classifier to calculate informativeness. In this case, the classifier must be applicable both in an unsupervised and a supervised setting. Second, some query strategies rely on labeled data, e.g., when estimating probability densities for the inlier class [15, 16]. In this case, one cannot calculate informativeness without labels. Current benchmarks mostly avoid this issue by simply assuming that some observations from each class are already labeled. In a real system, one must think about how to obtain the initially labeled observations [2, 21]. One option would be to start with a query strategy that does not require any label, such as random sampling, and switch to a more sophisticated strategy once there are sufficiently many labels. Another option is to let users pick the observations to label in the beginning, and then switch to an active learning strategy [2, 6]. However, deciding when to do switches between query strategies with OCAL is an open question.

Batch Query Selection Currently, query selection for one-class classifiers is sequential, i.e., for one observation at a time. However, this sequentiality may have several disadvantages, such as frequent updating and re-training of the one-class classifier. Further, it might be easier for users to label several observations in a batch than one observation at a time [34]. This may be the case when showing a diverse set of observations helps a user to develop an intuition regarding the data set. However, there currently are no strategies to select multiple observations in batches with one-class classifiers. An open question is whether strategies that have been proposed for other use cases, such as multi-class classification [12], are applicable with one-class classifiers.

Incremental Learning The runtime for updating a classifier constrains the frequency of querying the user. In particular, excessive runtimes for classifier training result in long waiting times and do away with interactivity. Intuitively, there is an upper limit that users are willing to wait, but the specific limit depends on the application.

Several strategies are conceivable to mitigate runtime issues. First, one can rely on incremental learning algorithms [20]. However, state-of-the-art one-class classifiers like SSAD [18] have been proposed without any feature for incremental learning. Second, one can sub-sample to reduce the number of training observations. Several strategies have been proposed explicitly for one-class classifiers [23, 26, 38]. But to our knowledge, there are no studies that combine sub-sampling with OCAL. Finally, one can use speculative execution to pre-compute the classifier update for both outcomes (inlier or outlier) while the user is deciding on a label [36]. While such a strategy requires additional computational resources, it might reduce waiting times significantly and improve interactivity. The open question is how to proceed with pre-computing when the look-ahead l is more than one feedback iteration. This is a combinatorial problem, and pre-computing all 2^l learning paths is intractable. Instead, one may use conditional probabilities to pre-compute only the most likely search paths. However, there currently is no method to plan pre-computation beyond $l = 1$. If users select observations to label by themselves, pre-computation would require to compute classifier update for all observations and outcomes, which is infeasible. Thus, there is a trade-off between giving users flexibility to decide freely on which observations to label, and the capabilities of pre-computation.

Evaluation at Runtime Without a good quality estimate, it is impossible to know whether the feedback obtained from a user already is sufficient [2], i.e., the one-classifier has converged, and additional feedback would not alter the decision boundary any further. However, evaluating the classification quality of OCAL at runtime is difficult [22]. This issue exists in both, when benchmarking with simulated feedback, and in real systems – here, we focus on the latter. Users may become frustrated if they face periods where their feedback does not have any effect.

However, showing users any estimated classification quality is difficult for two reasons. First, there might be a short term bias, i.e., the classifier performance might fluctuate significantly. This may be irritating, and it may be difficult to assess for the user. Second, the number of observations in the ground truth increases over time. With only a few labeled observations, the quality estimates may have a large error. This error may reduce with more iterations. So the open question is how to estimate classification quality reliably, and how to adapt these quality estimates during learning. One conceivable option is to switch between exploration and exploitation, i.e., switch from querying for examples that improve classification quality to selection strategies that improve the quality estimate of the classifier. However, there currently is no such switching method for OCAL.

Management of Data Flows Developing an active learning system also requires a sound software architecture. Although this is not a research challenge per se, there are several aspects to consider when implementing OCAL systems. One key aspect is the management of data flows. In particular, with a distributed application, see Section 2, there are several locations where one has to retain the data set, the classifier, the predictions, and the informativeness scores. For large data sets in particular, transferring data between a client and a backend or loading data sets from disc may affect runtimes significantly. This calls for efficient data caching. Further, one must decide where computations take place. For instance, to visualize contour plots, one must predict the decision boundary for a grid of observations, possibly in multiple projections of the data. In this case, transferring the model over the network may be very little overhead. This can be an efficient strategy when evaluating the model for an observation is cheap. This is the case with SVDD, since the model consists of only a few support vectors. With multi-user studies, one may even reuse trained classifiers and informativeness scores from other user sessions with an equivalent feedback history. In this case, it might be more efficient to pre-compute grid predictions in the backend. So there are several trade-offs and factors that determine an efficient data flow. There currently is no overview on these trade-offs. It also is unclear how they affect design decisions for OCAL systems.

4 Validating OCAL with User Studies

There are a few active learning user studies which have been conducted for special use cases, such as text corpus annotation [1, 11, 31] and network security [4]. However, it is unclear how findings relate to outlier detection with OCAL – the previous sections illustrate the peculiarities of this application. Further, the plethora of design options make user studies with OCAL systems particularly challenging.

Addressing all of the design options at once is not feasible, since there are too many combinations of classifiers, query strategies and ways to prepare information for users. So we propose to start with a narrow use case and to increase the complexity of the OCAL system step-wise. Specifically, we have identified the following steps towards a validation of OCAL in real applications.

(i) *Simplified Use Case:* Much of the value of active learning is in domains where obtaining labels is difficult, even for domain experts. However, we argue that one should identify a use case that many people can easily relate to. This has several advantages. First, we deem reproducibility more important than to obtain sophisticated insights on very special use cases. User studies are easier to reproduce when they do not depend on specific domain expertise. Further, when relationships in data are well understood, one can more easily judge whether the presentation of queries and results is accurate. So we argue to base a validation of OCAL on standard benchmark data, for instance the hand-written digit image data set MNIST[4]. Such a simplification also includes to fix the details of the feedback process, for instance to "sequential feedback" and "no initial labels". If necessary, one should downsample data sets so that runtimes of classifiers and query strategies are not a bottleneck.

(ii) *Validation of Information Presented:* The next step is to identify situations when users can give accurate feedback. Since the focus is to validate a learning system with users, one should start with a data set with available ground truth and select the best combination of classifier and query strategy in an experimental benchmark. This might seem counterintuitive at first sight. In a real application, there generally are not sufficiently many labels available to conduct such a benchmark – in fact, this may even be the motivation for active learning in the first place [2, 35]. However, we argue that this is a necessary step to break the mutual dependency between selecting a good setup and collecting labels. Given a combination of classifier and query strategy, one can then apply different query and result presentations and work with explanations and contextual information. By evaluating this step with user experiments, one can derive assumptions which, if met, enable users to provide accurate feedback.

(iii) *Validation of Classifier and Learning Strategy:* Based on these assumptions, one can vary the dimensions that have been fixed beforehand. This

[4] http://yann.lecun.com/exdb/mnist/

is, one fixes the information presented to the user and varies the query strategies and classifiers. Further, one may validate specific extensions such as batch query strategies.

(iv) *Generalization:* The first step of generalization is to scale the experiments to a large number of observations, using the techniques discussed in Section 3.5. Finally, one can then validate the approach on similar data sets, e.g., on different image data.

We expect the findings from these steps to be two-fold. On the one hand, we expect insights that are independent from the use case. For instance, whether scalability techniques are useful is likely to be use-case independent. On the other hand, many findings may depend on the type of data at hand. Explanations based on image data may be very different from the ones for, say, time series data.

Our OCAL system prototype already includes different classifiers and query strategies, see Section 2. So, in general, any researcher can already use our system to conduct Step (i) and the pre-selection of the query strategy and classifier information required for Step (ii). Regarding our prototype, the next steps are to select and implement a working set of query and result presentations, as well as to include black-box explainers and contextual information.

5 Conclusions

Validating One-Class Active Learning through user studies is challenging. One reason is that there are several open conceptual and technical challenges in the design and implementation of interactive learning systems. This article has featured a systematic overview of these challenges, and we have pointed out open research questions with one-class active learning. Next, we have sketched an architecture of a one-class active learning system, which we have implemented as a prototype. Based on it, we propose a roadmap towards validating one-class active learning with user studies.

Acknowledgement This work was supported by the German Research Foundation (DFG) as part of the Research Training Group GRK 2153: *Energy Status Data – Informatics Methods for its Collection, Analysis and Exploitation.*

References

1. Arora, S., Nyberg, E., Rosé, C.P.: Estimating annotation cost for active learning in a multi-annotator environment. In: NAACL Workshop. pp. 18–26. ACL (2009)
2. Attenberg, J., Provost, F.: Inactive learning?: Difficulties employing active learning in practice. SIGKDD Explor. Newsl. **12**(2), 36–41 (2011). https://doi.org/10.1145/1964897.1964906
3. Barnabé-Lortie, V., Bellinger, C., Japkowicz, N.: Active learning for One-Class classification. In: ICMLA. pp. 390–395. IEEE (2015). https://doi.org/10.1109/ICMLA.2015.167

4. Beaugnon, A., Chifflier, P., Bach, F.: End-to-end active learning for computer security experts. In: AAAI Workshop (2018)
5. Bernard, J., Hutter, M., Zeppelzauer, M., Fellner, D., Sedlmair, M.: Comparing visual-interactive labeling with active learning: An experimental study. Trans. Vis. Comput. Graph. **24**(1), 298–308 (2017)
6. Bernard, J., Zeppelzauer, M., Sedlmair, M., Aigner, W.: Vial: a unified process for visual interactive labeling. The Visual Computer **34**(9), 1189–1207 (2018)
7. Bischof, S., Trittenbach, H., Vollmer, M., Werle, D., Blank, T., Böhm, K.: Hipe: An energy-status-data set from industrial production. In: Proceedings of the Ninth International Conference on Future Energy Systems. pp. 599–603. ACM (2018)
8. Bögl, M., Filzmoser, P., Gschwandtner, T., Lammarsch, T., Leite, R.A., Miksch, S., Rind, A.: Cycle plot revisited: Multivariate outlier detection using a distance-based abstraction. Computer Graphics Forum **36**(3), 227–238 (2017)
9. Calma, A., Leimeister, J.M., Lukowicz, P., Oeste-Reiss, S., Reitmaier, T., Schmidt, A., Sick, B., Stumme, G., Zweig, K.A.: From active learning to dedicated collaborative interactive learning. In: ARCS 2016; 29th International Conference on Architecture of Computing Systems. pp. 1–8 (Apr 2016)
10. Cebron, N., Richter, F., Lienhart, R.: "I can tell you what it's not": active learning from counterexamples. Prog. Artif. Intell. **1**(4), 291–301 (2012). https://doi.org/10.1007/s13748-012-0023-9
11. Choi, M., Park, C., Yang, S., Kim, Y., Choo, J., Hong, S.R.: Aila: Attentive interactive labeling assistant for document classification through attention-based deep neural networks. In: CHI. ACM (2019). https://doi.org/10.1145/3290605.3300460
12. Demir, B., Persello, C., Bruzzone, L.: Batch-Mode Active-Learning methods for the interactive classification of remote sensing images. Trans. Geosci. Remote Sens. **49**(3), 1014–1031 (2011). https://doi.org/10.1109/TGRS.2010.2072929
13. Druck, G., Settles, B., McCallum, A.: Active learning by labeling features. In: EMNLP. pp. 81–90. ACL (2009)
14. Endert, A., Hossain, M.S., Ramakrishnan, N., North, C., Fiaux, P., Andrews, C.: The human is the loop: new directions for visual analytics. J. Intell. Inf. Syst. **43**(3), 411–435 (2014). https://doi.org/10.1007/s10844-014-0304-9
15. Ghasemi, A., Manzuri, M.T., Rabiee, H.R., Rohban, M.H., Haghiri, S.: Active one-class learning by kernel density estimation. In: MLSP Workshop. pp. 1–6 (2011). https://doi.org/10.1109/MLSP.2011.6064627
16. Ghasemi, A., Rabiee, H.R., Fadaee, M., Manzuri, M.T., Rohban, M.H.: Active learning from positive and unlabeled data. In: ICDM Workshop. pp. 244–250 (2011). https://doi.org/10.1109/ICDMW.2011.20
17. Görnitz, N., Kloft, M., Rieck, K., Brefeld, U.: Active learning for network intrusion detection. In: AiSec Workshop. ACM (2009). https://doi.org/10.1145/1654988.1655002
18. Görnitz, N., Kloft, M., Rieck, K., Brefeld, U.: Toward supervised anomaly detection. J. Artif. Intell. Res. **46**, 235–262 (2013)
19. Gupta, N., Eswaran, D., Shah, N., Akoglu, L., Faloutsos, C.: Beyond outlier detection: LOOKOUT for pictorial explanation. In: ECML. pp. 122–138. Springer (2018). https://doi.org/10.1145/1235
20. Kefi-Fatteh, T., Ksantini, R., Kaâniche, M.B., Bouhoula, A.: A novel incremental one-class support vector machine based on low variance direction. Pattern Recognit. **91**, 308–321 (2019). https://doi.org/10.1016/j.patcog.2019.02.027
21. Kottke, D., Calma, A., Huseljic, D., Krempl, G.M., Sick, B.: Challenges of reliable, realistic and comparable active learning evaluation. In: Proceedings of the Workshop and Tutorial on Interactive Adaptive Learning. pp. 2–14 (2017)

22. Kottke, D., Schellinger, J., Huseljic, D., Sick, B.: Limitations of assessing active learning performance at runtime. arXiv:1901.10338 (2019)
23. Krawczyk, B., Triguero, I., García, S., Woźniak, M., Herrera, F.: Instance reduction for one-class classification. Knowl. Inf. Syst. **59**(3), 601–628 (2019). https://doi.org/10.1007/s10115-018-1220-z
24. Legg, P., Smith, J., Downing, A.: Visual analytics for collaborative human-machine confidence in human-centric active learning tasks. Hum. Cent. Comput. Inf. Sci. **9**(1), 5 (2019). https://doi.org/10.1186/s13673-019-0167-8
25. Leite, R.A., Gschwandtner, T., Miksch, S., Kriglstein, S., Pohl, M., Gstrein, E., Kuntner, J.: Eva: Visual analytics to identify fraudulent events. Trans. Vis. Comput. Graph. **24**(1), 330–339 (2017). https://doi.org/10.1109/TVCG.2017.2744758
26. Li, Y.: Selecting training points for one-class support vector machines. Pattern Recognit. Lett. **32**(11), 1517–1522 (2011). https://doi.org/10.1016/j.patrec.2011.04.013
27. Lin, H., Gao, S., Gotz, D., Du, F., He, J., Cao, N.: Rclens: Interactive rare category exploration and identification. Trans. Vis. Comput. Graph. **24**(7), 2223–2237 (2017). https://doi.org/10.1109/TVCG.2017.2711030
28. Liu, S., Maljovec, D., Wang, B., Bremer, P.T., Pascucci, V.: Visualizing high-dimensional data: Advances in the past decade. Trans. Vis. Comput. Graph. **23**(3), 1249–1268 (2016). https://doi.org/10.1109/TVCG.2016.2640960
29. Phillips, R.L., Chang, K.H., Friedler, S.A.: Interpretable active learning. arXiv preprint arXiv:1708.00049 (2017)
30. Ribeiro, M.T., Singh, S., Guestrin, C.: “Why Should I Trust You?”: Explaining the predictions of any classifier. In: SIGKDD. pp. 1135–1144 (2016). https://doi.org/10.1145/2939672.2939778
31. Ringger, E.K., Carmen, M., Haertel, R., Seppi, K.D., Lonsdale, D., McClanahan, P., Carroll, J.L., Ellison, N.: Assessing the costs of machine-assisted corpus annotation through a user study. In: LREC. pp. 3318–3324 (2008)
32. Sacha, D., Zhang, L., Sedlmair, M., Lee, J.A., Peltonen, J., Weiskopf, D., North, S.C., Keim, D.A.: Visual interaction with dimensionality reduction: A structured literature analysis. Trans. Vis. Comput. Graph. **23**(1), 241–250 (2016). https://doi.org/10.1109/TVCG.2016.2598495
33. Seifert, C., Granitzer, M.: User-based active learning. In: ICDM Workshop. pp. 418–425. IEEE (2010). https://doi.org/10.1109/ICDMW.2010.181
34. Settles, B.: From theories to queries: Active learning in practice. In: AISTATS Workshop. pp. 1–18 (2011)
35. Settles, B.: Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning pp. 1–114 (2012)
36. Sperrle, F., Bernard, J., Sedlmair, M., Keim, D., El-Assady, M.: Speculative execution for guided visual analytics. In: VIS Workshop (2018)
37. Stokes, J.W., Platt, J.C., Kravis, J., Shilman, M.: Aladin: Active learning of anomalies to detect intrusions. Tech. rep., Microsoft Research (2008)
38. Sun, W., Qu, J., Chen, Y., Di, Y., Gao, F.: Heuristic sample reduction method for support vector data description. Turkish Journal of Electrical Engineering & Computer Sciences **24**(1), 298–312 (2016)
39. Tax, D.M.J., Duin, R.P.W.: Support vector data description. Mach. Learn. **54**(1), 45–66 (2004). https://doi.org/10.1023/B:MACH.0000008084.60811.49
40. Teso, S., Kersting, K.: Explanatory interactive machine learning. In: AAAI (2019)
41. Trittenbach, H., Englhardt, A., Böhm, K.: An overview and a benchmark of active learning for outlier detection with One-Class classifiers. arXiv:1808.04759 (2018)

42. Vollmer, M., Englhardt, A., Trittenbach, H., Bielski, P., Karrari, S., Böhm, K.: Energy time-series features for emerging applications on the basis of human-readable machine descriptions. In: Proceedings of the Tenth ACM International Conference on Future Energy Systems. pp. 474–481. ACM (2019)
43. Wilkinson, L.: Visualizing big data outliers through distributed aggregation. Trans. Vis. Comput. Graph. **24**(1), 256–266 (2017)
44. Yang, Y., Loog, M.: A benchmark and comparison of active learning for logistic regression. Pattern Recognit. **83**, 401–415 (2018). https://doi.org/10.1109/TVCG.2017.2744685
45. Yin, L., Wang, H., Fan, W.: Active learning based support vector data description method for robust novelty detection. Knowl. Based. Syst. **153**, 40–52 (2018). https://doi.org/10.1016/j.knosys.2018.04.020

RAL – Improving Stream-Based Active Learning by Reinforcement Learning

Sarah Wassermann[1], Thibaut Cuvelier[2], and Pedro Casas[1]

[1] AIT Austrian Institute of Technology – Vienna, Austria
sarah@wassermann.lu, pedro.casas@ait.ac.at
[2] CentraleSupélec – Gif-sur-Yvette, France; thibaut.cuvelier@supelec.fr

Abstract. One of the main challenges associated with supervised learning under dynamic scenarios is that of periodically getting access to labels of fresh, previously unseen samples. Labeling new data is usually an expensive and cumbersome process, while not all data points are equally valuable. Active learning aims at labeling only the most informative samples to reduce cost. In this paradigm, a learner can choose from which new samples it wants to learn, and can obtain the ground truth by asking an oracle for the corresponding labels. We introduce RAL – Reinforced stream-based Active Learning –, a new active-learning approach, coupling stream-based active learning with reinforcement-learning concepts. In particular, we model active learning as a contextual-bandit problem, in which rewards are based on the usefulness of the system's querying behavior. Empirical evaluations on multiple datasets confirm that RAL outperforms the state of the art, both by improving learning accuracy and by reducing the number of requested labels. As an additional contribution, we release RAL as an open-source Python package to the machine-learning community.

Keywords: Stream-based active learning · Reinforcement learning · Bandits

1 Introduction

A wide range of popular end-user applications such as Skype or Facebook Messenger request users' feedback about their experience at the end of a session. This user feedback is paramount for such services, as they may use the massively collected information to build prediction models shedding light on service performance, user engagement, preferences, etc. A major challenge when querying users is the fact that asking too often is annoying and might have negative consequences on engagement. In a general supervised-learning scenario, labeled data is extremely important, but labeling data is an expensive and tedious task, especially if based on human effort.

Active learning [9] offers a solution to the data exploration and exploitation trade-off, allowing a learner to interactively query the label of only the most informative samples. Active learning is generally used in an offline setup, to limit the number of labels required from a predefined set of unlabeled samples.

The learning tasks we tackle in this paper have two specific characteristics which are not covered by offline active learning: first, the learning data is time-and-size unbounded; second, it comes in the form of an online, non-periodic, sequential stream of samples. An appropriate learning approach should be able to track the evolution of the oracle feedback and the underlying concept drifts. In addition, it should do so by smartly and dynamically deciding at which specific times it is better to query the oracle. This would constantly improve the model-prediction capabilities and avoid unnecessarily querying the oracle.

Stream-based active-learning schemes have been proposed in the past, notably in [16,17]. However, as in traditional active learning, the decision on whether to request for a label or not is solely based on the the model's prediction uncertainty, and not on the informativeness of such a request. *The question that is raised is: can we improve the performance of stream-based active learning by considering how informative the requested label was?* We propose RAL – a novel Reinforced stream-based Active-Learning approach – combining both traditional stream-based active-learning techniques with reinforcement learning. While reinforcement learning has already been used to solve pool-based active-learning problems, the combination with stream-based active-learning algorithms is a mostly unexplored topic. RAL is specifically designed for stream-based, time-unbounded data, and, as we show next, it manages to improve the accuracy of the underlying supervised-learning model, while also significantly reducing the number of requested labels, compared to the state of the art. RAL is available as an open-source Python package[3].

2 Related Work

Many research efforts have already been undertaken in the field of active learning. For example, [16,17] present three simple approaches for this learning paradigm. Their proposed Randomized Variable Uncertainty approach tackles the problem of stream-based active learning using the model's prediction uncertainty to decide whether to query and trying to detect concept drifts by randomizing the certainty threshold used for querying decisions. [15] develops an active-learning algorithm with two different classifiers: one "reactive" and one "stable". The stable classifier is trained on all available labeled instances, while the reactive one learns on a window of recent instances. In [6], the authors present an active-learning technique based on clustering and prediction uncertainty. [7] conceives an approach relying on a modification of the Naïve Bayes classifier to update the different learners through the queried samples. In particular, the author uses one-versus-one classifiers to tackle multi-class problems and update the weights of the different classifiers by comparing their predictions to the ground truth. Krawczyk's technique behaves similarly to ours. However, the major difference is that he is using information about the classifiers' prediction certainty (without considering the corresponding weights) in order to adapt the minimum threshold

[3] `https://github.com/SAWassermann/RAL`

to query the oracle, while we rely on the usefulness of the decisions taken by RAL to tune the system according to the data stream.

Extending active learning through reinforcement learning is currently a very active research area. Active learning alone can easily converge on a policy of actions that have worked well in the past, but are sub-optimal. Reinforcement learning helps to improve the exploration-exploitation trade-off by letting the learner take risks with an uncertain outcome. However, most proposals do not consider the stream scenario, and operate on the basis of pool-based approaches. In [2,5], authors rely on the multi-armed bandit paradigm. [5] develops ALBL, which uses a modified version of EXP4 [1], a weight-updating rule, to attribute adaptive weights to different learners based on rewards; the learner to use is then determined through these weights; the chosen learner selects the samples in the pool to hand to the oracle based on its uncertainty measure. The approach described in [2] is similar to the one in [5], except for the reward-computation scheme. Some other papers in the field of pool-based active learning are [20,22]. The algorithm presented in [10,11] relies on the same principles as the system we are proposing, but tackles a different problem: Song's goal is to introduce an active-learning component into a contextual-bandit problem, while we are aiming at solving an active-learning problem by using contextual bandits.

Other recent papers dealing with active learning and reinforcement learning include [12,18,21,23]. However, most of them consider only one of the perspectives addressed by RAL, namely the enhancement of pool-based active learning through reinforcement learning [18,21,23], or the application of active learning to the streaming setup [12]. Combining active learning with reinforcement learning in a streaming, adaptive learning context is the most important contribution of RAL, a very timely yet vaguely addressed problem in the literature. [19,24] use the idea of *learning to active learn*, i.e. data-driven active learning. [19] proposes this view on pool-based active learning: the querying decision for a sample is based on an estimation of the accuracy improvement. [24] uses reinforcement learning in stream-based active one-shot learning, but this work is different from RAL on multiple aspects: (*i*) it tackles a different learning task, as it aims at detecting new classes instead of improving overall classification accuracy, (*ii*) their scheme relies on reinforcement learning only during the training phase and not once deployed, while RAL continuously adapts its querying policy during the whole incoming stream, and (*iii*) the system heavily relies on deep recurrent networks, too cumbersome to use in real-time resource-constrained scenarios, unlike RAL.

3 Introducing RAL

RAL relies on prediction uncertainty and reinforcement-learning principles, using rewards and bandit algorithms. The overall idea is summarized in Figure 1.

The intuition behind the different reward values is that we attribute a positive reward in case our system behaves as expected, and a negative one otherwise, to penalize it. RAL obtains rewards/penalties only when it is asking for ground truth. In a nutshell, it earns a positive reward ρ^+ in case it queries the oracle

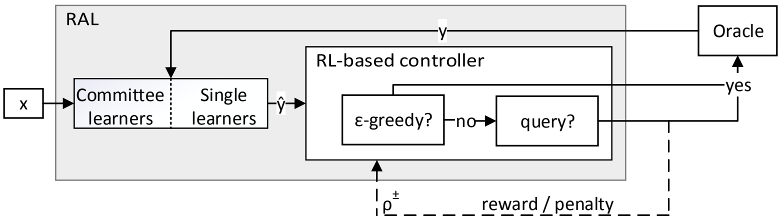


Fig. 1. Overall idea of the RAL system.

and would have predicted the wrong label otherwise (the system made the right decision to ask for the ground truth) and a penalty ρ^- (i.e. a negative reward) when it asks the oracle even though the underlying classification model would have predicted the correct label (querying was unnecessary). The rationale for using reinforcement learning is that RAL learns not only based on the queried samples themselves, but also from the usefulness of its decisions. The objective function to maximize is the total reward $\sum_{i=1}^{n} r_n$, where r_n is the nth reward (either ρ^+ or ρ^-) obtained by RAL.

The conceived system additionally makes use of the prediction certainty of the classification models. It is defined as the highest posterior classification probability among all possible labels for sample x. More formally, the certainty of a model is equal to $\max_{\hat{y}} P(\hat{y}|x)$ with $\hat{y}$ being one of all the possible labels for x.

The underlying assumption for designing RAL is based on the rationale that the model's uncertainty is an appropriate proxy for assessing the usefulness of a data point. Indeed, if the learner is uncertain about its prediction, this sample likely represents a region which has not been explored enough. Adding that sample with its true label to the training set would improve the overall prediction accuracy; the alternative is that the uncertainty is due to noise or to concept drift, these two points being especially challenging in a streaming setting. In RAL, we combine the aforementioned reward mechanism with the model's uncertainty to tune the sample-informativeness heuristic to better guide the query decisions.

Additionally, we implement an ε-greedy policy (also inspired by the bandit literature [14]) for the sake of data-space exploration: with probability ε, the system queries the oracle, even if the committee agreed not to query; we call this the *ε-scenario*. This ensures that we have a good chance of detecting potential concept drifts: without this policy, the system could end up being too confident about its predictions (and thus never ask the oracle again) even though its estimations are erroneous.

In the next two sections, we provide details about the single-classifier and the committee versions of RAL. We first devise the committee version, of which the single-classifier alternative is a simple adaptation.

3.1 Learning with a Committee of Learners

The algorithm behind RAL is summarized in Algorithm 1. Our approach is inspired by contextual bandits [1]. We rely on a set of experts (i.e. different

Algorithm 1 RAL algorithm.

```
procedure RAL($x$, E, $\alpha$, $\theta$, $\varepsilon$, $\eta$)
    $x$: sample to consider
    E: set of learners, members of the committee
    $\alpha$: vector of decision powers of learners in E
    $\theta$: certainty/querying threshold
    $\varepsilon$: threshold for $\varepsilon$-greedy
    $\eta$: learning rate for updating decision powers and the querying threshold
    decisions ← {}                                  ▷ will contain decisions of learners
    for e ∈ E do
        decisions[e] ← e.askCertainty($x$) < $\theta$          ▷ decisions[e] ∈ {0, 1}
    committeeDecision ← round$\left(\sum_{e \in E} \alpha[e] \cdot \text{decisions}[e]\right)$
    $p \leftarrow \mathcal{U}_{[0,1]}$                 ▷ random number drawn from a uniform distribution
    if $p < \varepsilon$ or committeeDecision = 1 then                  ▷ $\varepsilon$-scenario or not?
        $y$ ← acquireLabel($x$)
    if committeeDecision = 1 then
        $r$ ← getReward($x$, $y$)
        $\alpha$ ← updateDecisionPowers($r$, E, decisions, committeeDecision, $\alpha$, $\eta$)
        $\theta \leftarrow \min\left\{\theta\left(1 + \eta \times \left(1 - 2^{\frac{r}{\rho^-}}\right)\right), \quad 1\right\}$

function updateDecisionPowers($r$, E, decisions, committeeDecision, $\alpha$, $\eta$)
    for e ∈ E do
        if decisions[e] = committeeDecision then
            $\alpha[e] \leftarrow \alpha[e] \times \exp(\eta \times r)$                     ▷ EXP4
    $\alpha \leftarrow \alpha / \sum_{e \in E} \alpha[e]$                       ▷ normalize each value of $\alpha$
    return $\alpha$

function getReward($x$, $y$)
    return ($\rho^-$ if $\hat{y}(x) = y$ else $\rho^+$)
```

machine-learning models), referred to as a *committee*. Each expert gives its advice for the sample to consider: should the system ask the oracle for feedback or is the expert confident enough about its prediction? To assess a model's prediction certainty, we rely on a certainty threshold θ: if the model's certainty is below θ, the model is too uncertain about the prediction to make and thus it advises that RAL asks the ground truth. The query decision of the committee takes into account the opinions of the experts, but also their decision power: if the weighted majority of the experts votes for not querying, RAL will rely on the label prediction provided by the committee, used in the form of a voting classifier. The decision power of each expert gets updated such that the experts which agree with the entire committee are obtaining more power in case that particular decision is rewarding, i.e. informative (otherwise, these experts get penalized). These weights are updated through the EXP4 rule [1], with a learning rate η. RAL does not update the decision powers of the different learners in the ε-scenario: the committee did not take the querying decision and thus the weights of the models should not be impacted by this querying action.

The computation of the reward is carried out every time the committee decided to query (i.e. not in the ε-scenario). RAL therefore gets rewarded with ρ^{+} when it queried the oracle and asking was rewarding (i.e. the voting classifier would have predicted the wrong label). Conversely, RAL is penalized with ρ^{-} if the system used the oracle because the committee decided to do so, even though the underlying classifier would have predicted the correct class.

As an additional step, to ensure that RAL adapts as best as possible to the data stream, we do not only tune the weights of the committee members based on rewards, but also the uncertainty threshold θ, denoted in the remainder of this section as θ_n to stress that it is influenced by the $n-1$ samples observed so far. Again, as for the decision powers, θ_n is not updated in the ε-scenario.

The update rule of θ_n we implemented for our tool is written as follows:

$$\theta_n \leftarrow \min\left\{\theta_{n-1} \times \left(1 + \eta \times \left(1 - 2^{\frac{r_n}{\rho^-}}\right)\right), \quad 1\right\} \tag{1}$$

Design of update rule. In this subsection, we detail the reasoning behind our choice of the update policy. We are looking for a rule of this form:

$$\theta_n \leftarrow \min\left\{\theta_{n-1}\ (1 + f(r_n))\,, \quad 1\right\}, \tag{2}$$

where $f(r_n) = 1 - \exp(a \times r_n)$. The design goals of this update policy are that the threshold increases slightly when the reward is positive, conversely when the reward is negative. Our update policy should satisfy the following properties:

1. θ_n **should decrease *rapidly* in case r_n is negative**, as this indicates that the system queries too often and thus is performing poorly. Therefore, θ_n should be adapted fast to improve its performance.
2. θ_n **should *slightly* increase when r_n is positive**, so that the system does not keep decreasing the threshold. The model was right to ask for more samples, and thus the threshold should be increased. Nevertheless, the system is doing well: the threshold should not be too reactive to the queries.
3. f **must have two extrema**: a minimum at $\rho^{-} < 0$ and a maximum at $\rho^{+} > 0$.
4. θ_n **represents a probability**. $\theta_n = 0$ is not acceptable due to the product form of the update policy, thus the values of θ_n must be in the interval $(0, 1]$.
5. $f(r_n)$ **must be in the interval** $(-1, 1]$ to ensure that θ_n takes values corresponding to a probability. We exclude -1 from the allowed range of values to avoid that θ_n drops to 0.

The Properties 1 and 2 lead us to choose the family of functions $f_a : r \mapsto 1 - \exp(a \times r)$ parameterized by a. Property 5 can be translated into an equation to determine this parameter:

$$\lim_{r \to \rho^-} f(r) = 1 - \exp\left(a \times \rho^{-}\right) = -1 \tag{3}$$

After solving Equation 3, we get $a = \frac{\ln 2}{\rho^-}$. As f is strictly increasing, and because a is nonpositive, f will have a maximum when $r_n = \rho^{+}$. To satisfy Property 5, ρ^{+} must be chosen such that $f(\rho^{+}) \leq 1$.

As a final step, we introduce an additional hyperparameter to the update rule, namely the learning rate η. This rate aims at smoothing the evolution of the threshold θ_n, i.e. avoiding that θ_n changes too dramatically with a single query. We thus have the following update rule:

$$\theta_n \leftarrow \min\left\{\theta_{n-1}\left(1+\eta\times\left(1-2^{\frac{r_n}{\rho^-}}\right)\right), \quad 1\right\} \tag{4}$$

The values of η are restricted to the range $(0, 1)$. Indeed, we still must satisfy Property 5 (a value of 1 would violate this one) and $\eta = 0$ would lead to an nonreactive system, as the threshold would never adapt. Note that this version of RAL uses the same value of η for updating both θ_n and the decision powers in α.

Choice of hyperparameters. We acknowledge that RAL includes a non-negligible number of hyperparameters which should be well chosen in order to obtain the best results. While we do not have any rule of thumb on how to define exact values, the following guidelines help RAL learn from the streaming data:

- the initial value of θ should be high when the number of possible labels is low, to avoid that the model is always too certain about its prediction for the encountered samples
- ε should be higher when dealing with more dynamic datasets, to increase the probability to accurately grasp concept drifts; in general, we would advise using values in the range of 1 to 5%
- η should be small to avoid changing the decision powers of the different learners (α) and θ_n too abruptly; we would advise values below 0.1
- there is no specific range of values for $\rho^{\pm}$ which works better than others and these values should be picked considering the situation in which RAL is used: if unnecessary queries are a major issue, one should set ρ^- such that its absolute value is much higher than the one of ρ^+

3.2 Learning with a Single Classifier

RAL can also be used with a single classifier instead of a committee of learners. In that case, RAL becomes very lightweight and the only element of the system that allows it to efficiently adapt to and learn from the data stream is the variation of the uncertainty threshold θ by relying on the rewards r_n.

4 Expected Reinforcement Reward Analysis

As the main novelty of RAL lies in the introduction of a reinforcement learning loop to improve querying effectiveness and the data exploration-exploitation trade-off, we devote this section to the study of the reward properties in RAL. We rely on concepts from the bandit theory to understand its expected behavior. In the general case of a multi-class classification problem, under the assumptions that $\rho^+ \pm \rho^- \geq 0$, we prove the following bounds for RAL, $f_n(\alpha, \gamma)$ being a nonnegative function described next:

$$T\left(\rho^+ - \rho^-\right)\left[\alpha\, f_n(\alpha,\gamma) - 1\right] \leq \mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \leq \\ T\left(\rho^+ + \rho^-\right) + T\left(\rho^+ - \rho^-\right)\left[(1-\alpha)\, f_n(\alpha,\gamma) + \alpha\right] \quad (5)$$

We show in the experimental results (Section 5) that the cumulative reward is always positive for a very dynamic dataset, pointing to the good performance and added benefits of RAL.

4.1 Expected Reward Analysis – Single Classifier

Let us analyze the expected total reward obtained by using RAL, i.e. $\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\}$, where T denotes the number of samples in the considered data stream and r_n indicates the reward obtained for the nth sample.

In the following developments, we use these notations:

- $\hat{y}_n$ – nth predicted value
- $\hat{p}_n$ – certainty of the model for the nth prediction
- $\rho^{\pm}$ – reward and penalty obtained by RAL respectively; the reward must be nonnegative and the penalty nonpositive
- $\mathcal{VC}$ – VC dimension [13] of the learner
- θ_n – uncertainty threshold before having observed the nth sample
- err_n – error rate of our classifier before having observed the nth sample
- $\overline{err}_n$ – training error of model before having observed the nth sample

The expected total reward writes $\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} = \sum_{n=1}^{T} \mathbb{E}\left\{r_n\right\}$.

Based on the classical result of [13], we have the following bound:

$$f_n(\alpha) = \overline{err}_n + \sqrt{\frac{1}{N_n}\left[\mathcal{VC}\left(\log\frac{2N_n}{\mathcal{VC}} + 1\right) - \log\frac{\alpha}{4}\right]} \quad (6)$$

$$\mathbb{P}\left(err_n \leq f_n(\alpha)\right) = 1 - \alpha \quad (7)$$

where N_n denotes the training-set size for the underlying classifier at the nth round and α is a confidence level whose value lies in the interval $[0,1]$. We therefore can write this probabilistic bound as:

$$\mathbb{P}\Big[\mathbb{P}(\hat{y}_n \neq y_n) \leq f_n(\alpha)\Big] = 1 - \alpha \quad (8)$$

This means that the probability of making a mistake can be written as:

$$\begin{aligned}\mathbb{P}\left[\hat{y}_n \neq y_n\right] &= \mathbb{P}\left[\hat{y}_n \neq y_n | \mathbb{P}(\hat{y}_n \neq y_n) \leq f_n(\alpha)\right] \times \mathbb{P}\left[\mathbb{P}(\hat{y}_n \neq y_n) \leq f_n(\alpha)\right] \quad (9)\\ &+ \mathbb{P}\left[\hat{y}_n \neq y_n | \mathbb{P}(\hat{y}_n \neq y_n) > f_n(\alpha)\right] \times \mathbb{P}\left[\mathbb{P}(\hat{y}_n \neq y_n) > f_n(\alpha)\right]\\ &= \mathbb{P}\left[\hat{y}_n \neq y_n | \mathbb{P}(\hat{y}_n \neq y_n) \leq f_n(\alpha)\right](1-\alpha) + \mathbb{P}\left[\hat{y}_n \neq y_n | \mathbb{P}(\hat{y}_n \neq y_n) > f_n(\alpha)\right]\alpha\end{aligned}$$

Its upper and lower bounds are thus:

$$0 + \alpha\, f_n\,(\alpha) \leq \mathbb{P}\,[\hat{y}_n \neq y_n] \leq (1-\alpha)\; f_n\,(\alpha) + \alpha \times 1 \tag{10}$$

For the next proofs, we will require bounds on the probability of the certainty of the model being less than a threshold. Unfortunately, to the best of our knowledge, no generic result exists for the probability distribution of these certainties, which leads to very lose bounds:

$$0 \leq \mathbb{P}\,[\hat{p}_n \leq \theta_n] \leq 1 \tag{11}$$

For the following steps, we rely on classical results in probability theory, namely the union bound and Fréchet's inequality. For two probabilistic events A and B, be they independent or not, the following bounds hold:

$$\mathbb{P}(A \wedge B) \leq \mathbb{P}(A) + \mathbb{P}(B) \tag{12}$$

$$\mathbb{P}(A \wedge B) \geq \max\{0, \quad \mathbb{P}(A) + \mathbb{P}(B) - 1\} \tag{13}$$

We have that $\mathbb{E}\{r_n\} = \sum_{r\in\mathcal{R}} r \times \mathbb{P}(r_n = r)$ with $\mathcal{R} = \{\rho^+, \rho^-\}$ being the set of all possible reward values. As RAL does not obtain any reward in the ε-scenario, it can be ignored. Therefore, we have the following decomposition of the expectation and a generic upper bound:

$$\begin{aligned}
\mathbb{E}\{r_n\} = {} & \underbrace{\rho^+}_{\geq 0}\ \underbrace{\mathbb{P}\,[\hat{p}_n \leq \theta_n \wedge \hat{y}_n \neq y_n]}_{\leq (\mathbb{P}[\hat{p}_n \leq \theta_n] + \mathbb{P}[\hat{y}_n \neq y_n])} + \underbrace{\rho^-}_{\leq 0}\ \underbrace{\mathbb{P}\,[\hat{p}_n \leq \theta_n \wedge \hat{y}_n = y_n]}_{\geq (\mathbb{P}[\hat{p}_n \leq \theta_n] + \mathbb{P}[\hat{y}_n = y_n] - 1)} \\
& \leq \mathbb{P}\,[\hat{p}_n \leq \theta_n]\,\left(\rho^+ + \rho^-\right) + \mathbb{P}\,[\hat{y}_n \neq y_n]\ \rho^+ + [1 - \mathbb{P}\,[\hat{y}_n \neq y_n]]\ \rho^- - \rho^- \\
& \qquad \text{by factoring out the probabilities and using the opposite events} \\
& \leq \mathbb{P}\,[\hat{p}_n \leq \theta_n]\,\left(\rho^+ + \rho^-\right) + \mathbb{P}\,[\hat{y}_n \neq y_n]\,\left(\rho^+ - \rho^-\right)
\end{aligned} \tag{14}$$

Finally, the upper bound on the expected total reward, under the assumption that both $(\rho^+ + \rho^-)$ and $(\rho^+ - \rho^-)$ are nonnegative, is:

$$\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \leq T\left(\rho^+ + \rho^-\right) + T\left(\rho^+ - \rho^-\right)\ [(1-\alpha)\ f_n\,(\alpha) + \alpha] \tag{15}$$

If these two assumptions do not hold, a similar bound can still be achieved:

- First, suppose that $\rho^+ + \rho^- \geq 0$ and $\rho^+ - \rho^- \leq 0$. In this case, the only solution is to have $\rho^+ = \rho^- = 0$, thus trivially $\mathbb{E}\{\sum_{n=1}^{T} r_n\} = 0$
- Second, suppose that, conversely, $\rho^+ + \rho^- \leq 0$ and $\rho^+ - \rho^- \geq 0$. These assumptions lead to:

$$\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \leq T\left(\rho^+ - \rho^-\right)\ [(1-\alpha)\ f_n\,(\alpha) + \alpha] \tag{16}$$

- Third, the case where both $\rho^+ + \rho^- \leq 0$ and $\rho^+ - \rho^- \leq 0$ should not be studied further, because that would imply that $\rho^+ \leq 0$, which violates the defined range of allowed values for ρ^+ (in case $\rho^+ = 0$, we must have $\rho^- = 0$)

As a next step, we derive a lower bound of the expected total reward, with a very similar reasoning. First, the expected reward can be decomposed as:

$$\begin{aligned}\mathbb{E}\left\{r_n\right\} &= \underbrace{\rho^+}_{\geq 0}\ \underbrace{\mathbb{P}\left[\hat{p}_n \leq \theta_n \wedge \hat{y}_n \neq y_n\right]}_{\geq (\mathbb{P}[\hat{p}_n \leq \theta_n] + \mathbb{P}[\hat{y}_n \neq y_n] - 1)} + \underbrace{\rho^-}_{\leq 0}\ \underbrace{\mathbb{P}\left[\hat{p}_n \leq \theta_n \wedge \hat{y}_n = y_n\right]}_{\leq (\mathbb{P}[\hat{p}_n \leq \theta_n] + \mathbb{P}[\hat{y}_n = y_n])} \\ &\geq \mathbb{P}\left[\hat{p}_n \leq \theta_n\right]\ \left(\rho^+ + \rho^-\right) + \left(\mathbb{P}\left[\hat{y}_n \neq y_n\right] - 1\right)\ \left(\rho^+ - \rho^-\right) \\ &\text{by factoring out the probabilities and using the opposite events}\end{aligned} \tag{17}$$

Eventually, if $\rho^+ + \rho^- \geq 0$ and $\rho^+ - \rho^- \geq 0$, the expected total reward is at least $T\left(\rho^+ - \rho^-\right)\left[\alpha\, f_n\left(\alpha\right) - 1\right]$. Conversely, if $\rho^+ + \rho^- \leq 0$ and $\rho^+ - \rho^- \geq 0$, the lower bound is $T\left(\rho^+ - \rho^-\right)\left[\alpha\, f_n\left(\alpha\right) - 2\right]$.

4.2 Generalization to the multi-class case

The VC dimension makes no more sense when the classification problem includes multiple classes. There have been several generalizations thereof, for instance the covering number $\mathcal{N}^{(p)}(\gamma/4, \mathit{\Delta}_\gamma \mathcal{G}, 2\,N_n)$ [4], where $\mathit{\Delta}_\gamma \mathcal{G}$ is the set of classification margins obtained by any classifier of the family $\mathcal{G}$ in the known N_n data points (if a margin is larger than γ, it is clipped to γ). $\overline{err}_{\gamma,n}$ is the number of misclassifications, where an element is misclassified if its margin is less than γ. With a margin $\gamma \in \mathbb{R}_0^+$, a real number $\mathit{\Gamma} \in \mathbb{R}_0^+$ $(\gamma \leq \mathit{\Gamma})$, and the previously defined notations, the following bound on the generalization error holds:

$$f_n\left(\alpha, \gamma\right) = \overline{err}_{\gamma,n} + \frac{1}{N_n} + \sqrt{\frac{2}{N_n}\left[\log\left(2\,\mathcal{N}^{(p)}\Big(\frac{\gamma}{4}, \mathit{\Delta}_\gamma \mathcal{G}, 2\,N_n\Big)\right) - \log\frac{2\,\mathit{\Gamma}}{\alpha\,\gamma}\right]} \tag{18}$$

$$\mathbb{P}\left(err_n \leq f_n\left(\alpha, \gamma\right)\right) = 1 - \alpha \tag{19}$$

Notation is taken directly as defined in [4].

The upper bound of the expected total reward can be computed as in the binary-classification problem:

$$\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \leq T\left(\rho^+ + \rho^-\right) + T\left(\rho^+ - \rho^-\right)\ \left[\left(1 - \alpha\right)\ f_n\left(\alpha, \gamma\right) + \alpha\right] \tag{20}$$

Similarly, the lower bound for a multi-class problem can be expressed as:

$$\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \geq T\left(\rho^+ - \rho^-\right)\left[\alpha\, f_n\left(\alpha, \gamma\right) - 1\right] \tag{21}$$

4.3 Committee version

The mathematical developments for the committee version are very similar to the single classifier ones. First of all, the committee is still a classifier, and thus the same kind of bound applies on the probability of misclassifying. The only difference is that we have to take the VC dimension of the stacked classifiers instead of the one of the single classifier.

RAL asks the oracle for a label (and obtains the corresponding reward) if the weighted average of the decisions encourages it to query. We denote by $d_{i,n}$ the random variable indicating whether the ith classifier decides to query the oracle or not, i.e. whether its certainty $\hat{p}_{i,n}$ for the nth prediction is below the threshold θ_n (in case of querying, $d_{i,n} = 1$; otherwise, $d_{i,n} = 0$). $\alpha_{i,n}$ is the weight of the ith classifier for the nth sample; we have previously imposed that the sum of the weights must be one ($\sum_{i=1}^{C} \alpha_{i,n} = 1$ for each sample n). Thus, RAL asks when:

$$\sum_{i=1}^{C} \alpha_{i,n}\, d_{i,n} \geq \frac{1}{2} \tag{22}$$

For the upper bound, the previous developments still hold:

$$E\left\{r_n\right\} \leq \mathbb{P}\left[\sum_{i=1}^{C} \alpha_{i,n}\, d_{i,n} \geq \frac{1}{2}\right] \left(\rho^+ + \rho^-\right) + \mathbb{P}\left[\hat{y}_n \neq y_n\right] \left(\rho^+ - \rho^-\right) \tag{23}$$

Again, to the best of our knowledge, no generic result exists for a probability distribution of the querying decisions; we therefore have to resort to a very broad bound:

$$0 \leq \mathbb{P}\left[\sum_{i=1}^{C} \alpha_{i,n}\, d_{i,n} \geq \frac{1}{2}\right] \leq 1. \tag{24}$$

Finally, the expected total reward is, if $\rho^+ + \rho^- \geq 0$ and $\rho^+ - \rho^- \geq 0$, at most:

$$\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \leq T\left(\rho^+ + \rho^-\right) + T\left(\rho^+ - \rho^-\right)\ \left[(1-\alpha)\ f_n\left(\alpha,\gamma\right) + \alpha\right] \tag{25}$$

Similarly, concerning the lower bound, we obtain, for the same reasons, the same lower bound as in the single classifier case. Specifically, if $\rho^+ \pm \rho^- \geq 0$,

$$\mathbb{E}\left\{\sum_{n=1}^{T} r_n\right\} \geq T\left(\rho^+ - \rho^-\right)\left[\alpha f_n\left(\alpha,\gamma\right) - 1\right] \tag{26}$$

4.4 Summary

This theoretical analysis of RAL yields bounds on the expected total reward which could be tightened by stronger results on the probability distribution of $\hat{p}_n$. Nevertheless, our results show that the expected total reward is *significantly higher* than $T\rho^-$, whatever the values of $\rho^\pm$: RAL usually takes the appropriate decision, and thus mostly queries when necessary. Conversely, the upper bound is always nonnegative.

	MAWI	Woodcover
ε	2.5%	5%
η	0.01	0.02
initial θ	0.9	0.9
ρ^+	1	1
ρ^-	-1	-1

Fig. 2. RAL hyperparameters, selected by grid search.

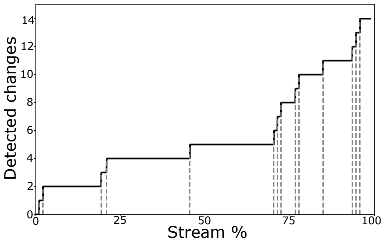


Fig. 3. Concept-drift detection in MAWI. Changes are marked with dashed lines.

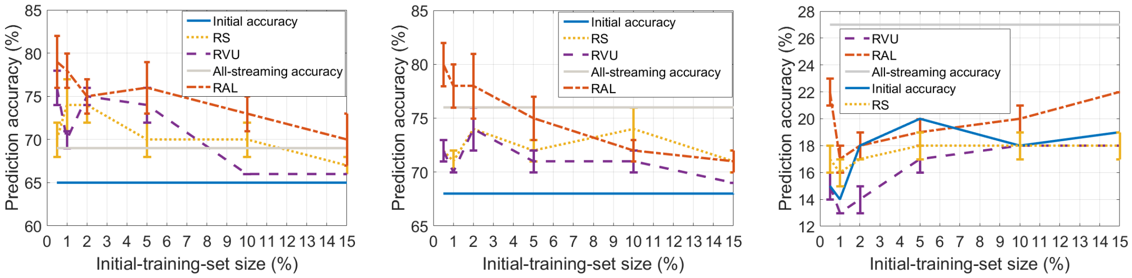
It is more beneficial to choose the rewards such that $\rho^+ + \rho^- \geq 0$. Indeed, in this case, the upper bound is higher (we add a term $T(\rho^+ + \rho^-)$ to the initial bound) as well as the lower bound (this time, we add a term $T(\rho^+ - \rho^-)$). This means that, for a promising behavior of RAL, good decisions should be more rewarded than bad ones are penalized.

By studying special cases of these formulae, we can obtain significant insight into their properties. For instance, if the prediction algorithm is very inaccurate (a situation that is expected for the first samples of the stream) and almost constantly infers the wrong class, i.e. if $\alpha f_n(\alpha, \gamma)$ is close to 1 (equal to $1 - \zeta$), the lower bound becomes $T(\rho^+ - \rho^-)\zeta$. This means that the expected total reward, in this case, is at least zero. This result is significantly stronger than the trivial bound $T\rho^-$: RAL's decisions generate a positive total reward, on average.

5 Evaluation

To showcase the performance of RAL, we evaluate our tool and compare it to a state-of-the-art algorithm for stream-based active learning, and to random sampling (RS). In particular, we compare RAL to the Randomized Variable Uncertainty (RVU) technique proposed in [16,17] and mentioned in Section 2, as this approach also heavily relies on the uncertainty of the underlying machine-learning model to take the querying decisions. We use three datasets for the sake of generalization, namely two datasets extracted from MAWILab [3] and a subset of the widely used Forest Covertype data[4]. The covertype dataset contains samples labeled with forest cover type represented by cartographic variables. The dataset provided by MAWILab is publicly available and consists of 15-minute-network-traffic traces captured every day on a backbone link between Japan and the US since 2001 in a stream-based fashion. Traces are annotated with the attack types observed during their corresponding measurement period. In this work, we focus on two attack detections, namely the flood and the UDP-netscan attacks. The MAWI data is subject to concept drifts. We relied on a commonly used statistical test, namely, the Page-Hinkley test [8], for the detection of changes. Figure 3 depicts the cumulative number of drifts observed in the dataset. The test detects 14 abrupt changes during the total measurement time span.

[4] `https://archive.ics.uci.edu/ml/datasets/Covertype`, accessed in April 2019

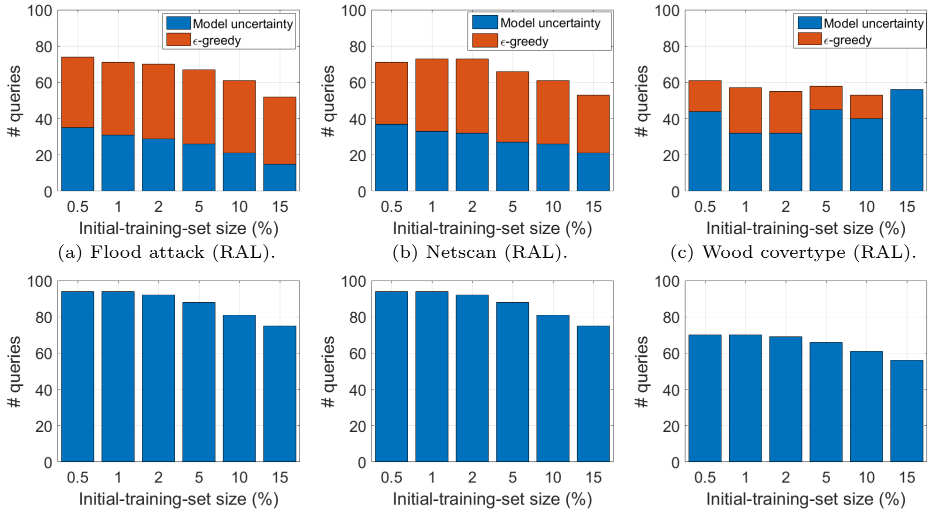


(a) Flood attack.

(b) Netscan.

(c) Wood covertype.

Fig. 4. Prediction-accuracy evaluation for RAL, RVU, and RS. For each of the tested datasets, RAL outperforms both techniques.



(a) Flood attack (RAL).

(b) Netscan (RAL).

(c) Wood covertype (RAL).

(d) Flood attack (RVU).

(e) Netscan (RVU).

(f) Wood covertype (RVU).

Fig. 5. Number of queries issued by RAL and RVU. RAL asks for significantly fewer samples to reach a better accuracy than RVU.

5.1 Setup

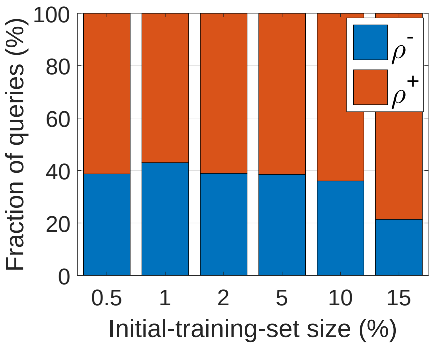
For each benchmarked algorithm, we proceed as follows: first, we subdivide the considered datasets into three consecutive, disjoint parts, i.e. the initial training set, the streaming data, and the validation set. The validation set consists of the last 30% of the dataset, the initial training set is a variable fraction of the first samples (varying between the first 0.5%, 1%, 2%, 5%, 10%, and 15%), and the streaming part includes all the remaining samples not belonging to the other two subsets. We then train a model on the initial training set and check its accuracy (referred to as the *initial accuracy*) on the validation part. Next, we run the benchmarked algorithm on the streaming part and let it pick the samples it wants to learn from. We retrain the models after each queried label. Finally, we evaluate the final model (i.e. trained on the initial training set and the chosen samples) again on the validation set and report this model's prediction accuracy (referred to as the *final accuracy*). In the context of this evaluation, we implement for both RAL and RVU the budget mechanism presented in [16], based on the ratio between the number of queries and the total number of samples observed so far; the system is allowed to issue queries to the oracle as long as this ratio is below a certain threshold, i.e. the budget. For random sampling, we use a budget indicating the exact number of samples to ask feedback for. For each dataset, we set it to the highest average number of queried samples by either RAL or RVU among all the tests with all the considered initial-training-set sizes. All tests are repeated 10 times, and we report both the average accuracy and its standard error. For RAL, we indicate the average number of queries performed due to the uncertainty of the underlying model and the ones issued through the ε-greedy mechanism. For RVU, we also report the average number of queries. The hyperparameter values of RAL are chosen by grid search for our datasets on the training set within the ranges prescribed in Section 3. The used values are indicated in Figure 2. For RAL and RVU, the budget is set to 0.05 for all the experiments and we test RVU with the hyperparameters recommended in [17].

5.2 Results

The results are shown in Figures 4 and 5. The reported *all-streaming accuracy* refers to the accuracy obtained by the model in case it queries all the samples seen in the stream.

For the evaluations based on the two MAWI datasets, we use the committee version of RAL. More precisely, the committee is a voting classifier composed of a k-NN model with k equal to 5, a decision tree, and a random forest with 10 trees. We also use the same model for RVU and RS. On Figures 4(a) and (b), we can clearly note that RAL outperforms both RVU and RS on average. A striking example are the results for the netscan detection, where RAL obtains final accuracies which are 5 percent points higher than the ones of RVU and RS for the two smallest initial-training-set sizes. It is also worth highlighting that RAL is the only algorithm yielding higher final accuracies than the all-streaming one as long as the initial training set contains less than 5% of the data. To our surprise, RVU is often outperformed by RS. Finally, the flood-attack-detection analysis shows that the three approaches often yield a final accuracy higher than the all-streaming one, underlining that learning from the entire data stream does not necessarily output the best possible accuracy. When it comes to the number of queried samples, we see that RAL queries on average significantly less often than RVU, and a non-negligible part of these queries are due to the model's uncertainty, suggesting that the samples picked by RAL for its learning purposes are wisely chosen.

For the evaluation on the Forest Covertype dataset, we rely on the single-classifier version of RAL, using a 10-tree random forest. As for MAWI, RVU and RS use the same model as RAL. Again, RAL yields better final accuracies than both RVU and RS, even though this prediction task is very challenging.



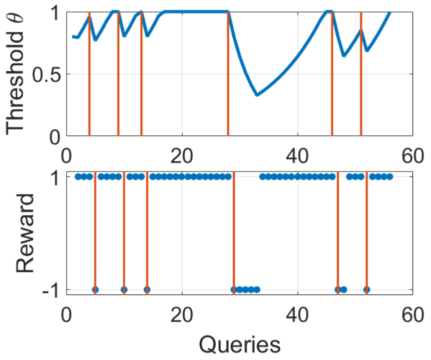


Fig. 6. Proportion of obtained rewards vs. obtained penalties by RAL – wood covertype dataset.

Fig. 7. Evolution of RAL's uncertainty threshold w.r.t. rewards – wood covertype dataset. Red lines indicate penalties.

Moreover, in this study, RVU performs at most as well as RS, but generally worse, showing that the decision-making algorithm of RVU is not always appropriate for complex tasks. Next, we analyze the ratio between the rewards obtained by RAL versus its penalties, i.e. we check whether querying the oracle when uncertain is necessary. Figure 6 shows the outcome of this analysis and it is very encouraging. Indeed, for each test case, the fraction of useful queries is at least 60%. Lastly, we analyze the reactivity of RAL's certainty threshold θ with respect to the obtained rewards and penalties. Figure 7 depicts the evolution of θ with respect to the rewards while RAL is processing the data stream. θ reacts swiftly to penalties, i.e. its value decreases rapidly once RAL gets penalized and, very often, the next query is useful (reward equal to ρ^+), underlining the fact that updating the threshold based on rewards is very relevant.

Note that the initial accuracy is constant for the two different MAWILAB datasets. This is due to the fact that the first 15% of these datasets consist of points with the same label (more precisely, they represent an attack). The first parts of the woodcover dataset are much more dynamic.

6 Conclusions

We have introduced RAL, a novel Reinforced stream-based Active-Learning approach, to tackle challenges of stream-based active learning, i.e. selecting the most valuable sequentially incoming samples, using reinforcement-learning principles. It does not only learn from the data stream, but also from the relevance of its querying decisions. RAL provides a completely different exploration-exploitation trade-off than existing algorithms, as it queries fewer samples of higher relevance. The theoretical analysis underlines the encouraging behavior of RAL. We showed on several datasets that RAL provides promising results, outperforming the state of the art. We make RAL publicly available as an open-source Python package.

References

1. P. Auer et al.: The Nonstochastic Multiarmed Bandit Problem. SIAM Journal on Computing, vol. 32(1), pp. 48–77, 2002
2. Y. Baram et al.: Online Choice of Active Learning Algorithms. Journal of Machine Learning Research (JMLR), vol. 5, pp. 255–291, 2004
3. R. Fontugne et al.: MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking. ACM CoNEXT, 2010
4. Y. Guermeur: VC Theory of Large Margin Multi-Category Classifiers. Journal of Machine Learning Research (JMLR), vol. 8, pp. 2551–2594, 2007
5. W.N. Hsu et al.: Active Learning by Learning. AAAI Conference on Artificial Intelligence (AAAI), 2015
6. D. Ienco et al.: Clustering Based Active Learning for Evolving Data Streams. Discovery Science. pp. 79–93, 2013
7. B. Krawczyk: Active and adaptive ensemble learning for online activity recognition from data streams. Knowledge-Based Systems, vol. 138, pp. 69–78, 2017
8. E.S. Page: Continuous Inspection Schemes. Biometrika, vol. 41, pp. 100–115, 1954
9. B. Settles: Active Learning Literature Survey. Tech. rep., University of Wisconsin-Madison, 2010
10. L. Song: Stream-based Online Active Learning in a Contextual Multi-Armed Bandit Framework. arXiv:1607.03182, 2016
11. L. Song et al.: A Contextual Bandit Approach for Stream-Based Active Learning. arXiv:1701.06725, 2017
12. A. Santoro et al.: One-shot Learning with Memory-Augmented Neural Networks. arXiv:1605.06065, 2016
13. V. Vapnik: The Nature of Statistical Learning Theory, 2000
14. C. Watkins: Learning from Delayed Rewards. Ph.D. thesis, King's College, Cambridge, 1989
15. W. Xu et al.: Active Learning Over Evolving Data Streams Using Paired Ensemble Framework. International Conference on Advanced Computational Intelligence (ICACI), 2016
16. I. Žliobaitė et al.: Active Learning with Evolving Streaming Data. Machine Learning and Knowledge Discovery in Databases. pp. 597–612, 2011
17. I. Žliobaitė et al.: Active Learning With Drifting Streaming Data. IEEE Transactions on Neural Networks and Learning Systems, vol. 25(1), pp. 27–39, 2014
18. P. Bachman et al.: Learning Algorithms for Active Learning. International Conference on Machine Learning (ICML), 2017
19. K. Konyushkova et al.: Learning Active Learning from Real and Synthetic Data. Conference on Neural Information Processing Systems (NIPS), 2017
20. G. Contardo et al.: A Meta-Learning Approach to One-Step Active Learning. International Workshop on Automatic Selection, Configuration and Composition of Machine Learning Algorithms, 2017
21. M. Fang et al.: Learning how to Active Learn: A Deep Reinforcement Learning Approach. Conference on Empirical Methods in Natural Language Processing (EMNLP), 2017
22. S. Ravi et al.: Meta-Learning for Batch Mode Active Learning. International Conference on Learning Representations (ICLR), Workshop Track, 2018
23. K. Pang et al.: Meta-Learning Transferable Active Learning Policies by Deep Reinforcement Learning. International Conference on Machine Learning (ICML), AutoML Workshop, 2018
24. M. Woodward et al.: Active One-shot Learning. Conference on Neural Information Processing Systems (NIPS), Deep Reinforcement Learning Workshop, 2016

Knowledge-based Selection of Gaussian Process Surrogates

Zbyněk Pitra[1,2], Lukáš Bajer[1,3], and Martin Holeňa[1]

[1] Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic
{pitra, holena}@cs.cas.cz
[2] Faculty of Nuclear Sciences and Physical Engineering, CTU in Prague
Břehová 7, 115 19 Prague 1, Czech Republic
[3] Cisco Systems, Czech Republic
Karlovo nám. 10, 120 00 Prague, Czech Republic
bajeluk@gmail.com

Abstract Many real-world problems belong to the area of continuous black-box optimization. If the black-box function is also cost-aware, regression surrogate models are often utilized by optimization algorithms to save evaluations of the original cost-aware function. Choosing a suitable surrogate model or a suitable setting of its hyperparameters is a complex selection problem, where research into reusing knowledge represented by features of black-box function landscape is only starting. In this paper, we report the research into surrogate model selection, where knowledge from the previous experience with using the model is utilized to design a metalearing system. As a proof of concept, we provide a study investigating the influence of landscape features on the performance of various Gaussian process covariance functions as surrogate models for the state-of-the-art optimization algorithm in the cost-aware continuous black-box optimization.

Keywords: Benchmarking · Black-box optimization · Gaussian process · Landscape analysis

1 Introduction

Surrogate modeling is a technique for saving expensive evaluations of a black-box objective function during the run of an optimization algorithm. Given a set of observations, a surrogate model can be fitted to approximate the landscape of the objective function. However, which surrogate model should be chosen given a particular optimization task? Generally, no surrogate model improves the algorithm always better than all other surrogate model approaches (cf. [14,28]). The performance of each surrogate-assisted algorithm obviously depends on the properties of the data; therefore, investigation of the suitability of different models and their settings for different combinations of the data properties is very much needed.

Surrogate model selection can utilize the experience from the application of the considered models to other optimization tasks, a strategy known as *metalearing* [19]. Considering the surrogate model selection problem, it is necessary to extract information about the approximated function, which can be later utilized by a learning system to make a decision about the convenience of particular surrogate models. Therefore, features characterizing properties of the landscape of the objective function should help to better distinguish the model suitability.

In recent years, many features aiming to describe the properties of objective function landscapes have been proposed (cf. the overview in [16]). However, a majority of landscape features was utilized only for the selection of optimization algorithms and algorithm settings (a. k. a. *Algorithm Selection* or *Algorithm Configuration* problems [33]), not for the selection of surrogate models and their settings. The discussion in [14] suggests that landscape features can be used to this end, too. However, only little research in that direction is known so far.

In this paper, we report a research into designing a metalearing system for surrogate model selection according to past experience. We study relations between the performance of surrogate models and considered properties of objective function landscapes. As a proof of concept, we utilize results of the investigation in [29], where the influence of Gaussian process (GP) covariance function settings on the error of GP predictions with respect to the original fitness has been studied in connection with landscape features. We employ a classification tree showing the dependence of the most suitable covariance function on landscape features to adaptively select the most promissing covariance for the GP surrogate model in the surrogate variant of the state-of-the-art black-box optimization algorithm Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [10], the Doubly Trained Surrogate CMA-ES (DTS-CMA-ES). We evaluate the resulting algorithm performing automatical covariance function selection on the noiseless part of the COCO framework [11,12] and compare it to five DTS-CMA-ES versions without online covariance function selection.

The next section provides a brief introduction to surrogate modeling and landscape analysis. Section 3 states the proposed research problem and our approach to adress it. Section 4 presents a proof of concept of the proposed approach and its experimental results. The last section discusses the results and suggests directions for future research.

2 Background

2.1 Surrogate Modeling

Replacing an expensive function f with a trained regression model has been used to speed-up black-box optimization for many years. Such regression model, a. k. a. *surrogate model*, is trained on the already available input–output value pairs $(\mathbf{x}_i, y_i), i = 1, \ldots, N$, where $\mathbf{x}_k$ is a point in a search space and $y_k = f(\mathbf{x}_k)$ is an objective function value of x_k for $k = 1, \ldots, N$. The model is used instead of the original expensive objective function to evaluate some of the points needed by the optimization algorithm. The *response-surface models* [26] are low-degree polynomial models and were used as the historically first models in costly continuous optimization [1,15]. Since then, other models like multi-layer perceptron- and RBF-networks [34], support vector machine regression [20], random forests [2] or Gaussian processes [2,5,27,35] were also used in black-box optimization.

Simpler models like polynomials are cheap to train; they are thus suitable for the applications where additional computational resources imposed by the model building would constitute a substantial part of the overall optimization cost. On the other hand, random forests and Gaussian processes provide estimation of the prediction uncertainty which can be used in selecting points for evaluation either with the expensive original function, or with the model fitness function [3,27].

2.2 Landscape Analysis

Landscape analysis aims at characterizing the landscape of an investigated function and deriving rules how those characteristics influence the performance of the optimization algorithm. The final goal is to formulate rules for the selection of suitable algorithms for an unknown problem according to the calculated features; this corresponds to the *Algorithm Selection* problem formulated in [33].

A large number of various landscape analysis techniques have been proposed in recent years. The following measures quantifying the characteristics of landscapes were formulated in [23]: *multi-modality*, *global structure*, *separability*, *variable scaling*, *search space homogeneity*, *basin size homogeneity*, *plateaus*, and *local to global optima contrast*. However, the majority of these *high-level properties* have the disadvantages of expert knowledge necessity, categorical character, missing important information, and requiring knowledge about the whole problem [16].

Exploratory Landscape Analysis [22] is an umbrella term for all such methods, even though originally developed for combinatorial optimization problems [25]. An important step in the development of landscape analysis was a proposal of six *low-level* easy to compute feature classes [22], each containing a number of individual features. Generally in continuous black-box optimization field, such feature is a function $\varphi : \bigcup_{N \in \mathbb{N}} \mathbb{R}^{N,D} \times \mathbb{R}^{N,1} \mapsto \mathbb{R}$ which aims to describe landscape properties utilizing a dataset of N pairs of observations $\left\{(\mathbf{x}_i, y_i) \in \mathbb{R}^D \times \mathbb{R} \,|\, i = 1, \ldots, N\right\}$. Proposed feature classes represent measures related to the distribution of the objective function values (*y-Distribution*), the relative position of each value with respect to quantiles (*Levelset*), the information extracted from linear or quadratic regression models fitted to the sampled data (*Meta-Model*), and three feature classes requiring additional objective function evaluations – the level of convexity (*Convexity*), gradient and Hessian approximation statistics (*Curvature*), and features related to local searches conducted from sampled points (*Local Search*). It was shown [22] that these low-level features relate well the above mentioned high-level properties.

The *cell-mapping* approach [18] discretizes the input space to a user-defined number of blocks (i. e., cells) per dimension. Afterwards, the corresponding features are based on the relations between the cells and points within. Three cell-mapping feature classes were defined: features extracting information based on the location of the best and worst observation within a cell w.r.t. the corresponding cell center, aggregated cell-wise information on the gradients between each point of a cell and its corresponding nearest neighbor, and estimated convexity of representative observations from three successive cells in each dimension. Additionally, the *Generalized Cell Mapping* features are based on estimated transition probabilities of moving from one cell to one of its neighboring cells. Using those probabilities, the *barrier tree* [7] can be constructed to represent the local optima by tree leaves and landscape ridges by the branching nodes. It should be noted that cell-mapping approach is less useful in higher dimensions where majority of cells is empty and feature computation can require a lot of time and memory.

Nearest better clustering (*NBC*) features [17] are based on the detection of funnel structures. The calculation of such features is based on the comparison of distances from observations to their nearest neighbors and their *nearest better neighbors*, which are the nearest neighbors among the set of all observations with a better objective value. In [21], the set of *dispersion features* comparing the dispersion among the data points and among subsets of these points from the dataset is proposed. The *information content* features of a continuous landscape are derived in *Information Content of Fitness Sequences* approach [24] as the adaptation of methods for calculating of the information content of discrete landscapes. In [16], three feature sets were proposed: the features providing basic information about the data such as the number of points, boundaries or dimension (*Basic*), aggregated information about coefficients of linear models fitted in each cell, and information obtained from *principle component analysis* measuring the proportion of principle components needed to explain a user-defined percentage of variance. A comprehensive survey of landscape analysis methods can be found, e. g., in [25].

Research into using landscape features for surrogate modeling selection has started only recently. In [36], the *fitness distance correlation* was utilized for automatic selection between polynomial and RBF models and their settings as surrogates for a particle swarm optimization algorithm. In [30], we have investigated relationships between two surrogate models (GP and RF) and a set of landscape features. In [29], we have proposed the set of landscape features based on the state variables of the CMA-ES algorithm (*CMA features*) and investigated the relationships of GP covariance functions to landscape features.

3 Landscape Analysis for Surrogate Model Selection

3.1 Surrogate Model Selection Problem

The surrogate model selection problem can be formalized as follows: In an iteration i of a surrogate-assisted algorithm A, a set of surrogate models $\mathcal{M}$ with hyperparameters $\boldsymbol{\theta}$ are trained utilizing particular choices of the training set $\mathcal{T}$. The training set $\mathcal{T}$ is selected out of an *archive* $\mathcal{A}$ ($\mathcal{T} \subset \mathcal{A}$) using some training set selection method (*TSS*). The archive contains all points in which the fitness f has been evaluated so far $\mathcal{A} = \{(\mathbf{x}_i, f(\mathbf{x}_i)) | \, i = 1, \ldots, N\}$. Afterwards, the surrogate model $M \in \mathcal{M}$ is utilized to evaluate new set of points (*population*) $\mathcal{P} = \{\mathbf{x}_k | k = 1, \ldots, \alpha\}$, where $f(\mathbf{x}_k)$ can be obtained using the expensive black-box fitness function and $\alpha \in \mathbb{N}$ depends on the strategy for the selection of new points for evaluation by the models from $\mathcal{M}$. The main question related to this problem is: How can we select the most convenient models from the set $\mathcal{M}$ (and possibly $\boldsymbol{\theta}$) according to $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{P}$?

3.2 Proposed Methodology

We suggest to use the metalearing approach based on landscape features to tackle the surrogate model selection problem.

Learning phase: First, a set of datasets $\mathcal{D} = \{\mathcal{A}^{(l)}, \mathcal{T}^{(l)}, \mathcal{P}^{(l)}\}_{l=1}^{L}$, $L \in \mathbb{N}$, is created (ideally via recording the datasets from independent runs of the algorithm A). Second, for each l, each model $M \in \mathcal{M}$ with hyperparameters $\boldsymbol{\theta}_M$ is trained on $\mathcal{T}^{(l)}$ and its performance is assessed with some error measure ε on $\mathcal{P}^{(l)}$. Third, each dataset from $\mathcal{D}$ is characterized using a set of landscape features Φ. In this way, a mapping $S_M : \Phi \to \mathcal{M}$ or $S_\theta : \Phi \to \bigcup_{M \in \mathcal{M}} \boldsymbol{\Theta}_M$ from feature space to $\mathcal{M}$ or $\bigcup_{M \in \mathcal{M}} \boldsymbol{\Theta}_M$ is learned, where $\boldsymbol{\Theta}_M$ stands for the set of possible hyperparameters of the model M.

Application phase: In each iteration i of an algorithm A, the landscape features Φ are calculated on datasets $\mathcal{A}^{(i)}, \mathcal{T}^{(i)}, \mathcal{P}^{(i)}$. After that, the mapping S is used to select the surrogate model $M \in \mathcal{M}$ and its hyperparameters $\boldsymbol{\theta}_M \in \boldsymbol{\Theta}_M$. The selected $M \in \mathcal{M}$ is trained on $\mathcal{T}^{(i)}$ and then utilized for predicting fitness values of the elements of $\mathcal{P}^{(i)}$.

4 Proof of Concept

4.1 Learning phase

Optimization Algorithm Considering cost-aware black-box single-objective optimization of continuous functions, the CMA-ES [10] has been many times successfully improved using surrogate models to save fitness function evaluations [13,15,20,27]. The DTS-CMA-ES [3,27] has been shown a valuable representative of such surrogate-assisted versions of the CMA-ES. Therefore, we have utilized DTS-CMA-ES to play the role of the algorithm A in our concept.

Surrogate Model and Hyperparameters As a surrogate model, the DTS-CMA-ES uses Gaussian processes [31] due to their ability to estimate the whole distribution of the fitness function. In the DTS-CMA-ES, the Gaussian process model setting is fixed during the whole optimization process, so is the GP covariance function. An essential GP hyperparameter is the type of covariance function. In [32], we have proposed to select the covariance function for a GP-based surrogate model for the CMA-ES using a Bayesian approach.

Mapping The results in [29] suggested that mapping from the space of features calculated on $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{P}$ to the value set of a categorical hyperparameter can be represented by a classification tree.

Error Measure The CMA-ES state variables are adjusted according to the ordering of μ best points from the current population. Therefore, the Ranking Difference Error [3] is a convenient measure of model error for the DTS-CMA-ES

$$RDE_{\mu}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\sum_{i:(\rho(\mathbf{y}))_i \leq \mu} |(\rho(\mathbf{y}))_i - (\rho(\hat{\mathbf{y}}))_i|}{\max_{\pi \in \text{Permutations of } (1,\ldots,\lambda)} \sum_{i:\pi(i) \leq \mu} |\, i - \pi(i)|}\,, \tag{1}$$

where $(\rho(\mathbf{y}))_i$ is the rank of y_i among the components of $\mathbf{y}$.

Dataset To generate a set of datasets $\mathcal{D}$, we have used independent runs of the DTS-CMA-ES on the 24 noiseless single-objective benchmarks from the COCO framework [11,12] in dimensions 2, 3, 5, 10, and 20 on instances 11–15. Using each of the 8 different covariance functions from [29] in each of those independent runs, data from 25 uniformly selected generations were recorded. The runs of the algorithm were terminated in cases when the limit of 250 function evaluations per dimensions was exceeded or when the target fitness value 10^{-8} was reached. The details[1] of generating the datasets can be found in [29].

Landscape Features The following 6 feature classes were employed to characterize all the sets $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{P}$ from the datasets in $\mathcal{D}$: *y-Distribution*, *Levelset*, *Meta-Model*, *NBC*, *Dispersion*, *Information Content*, and *CMA features*. In addition, the *dimension D* and the *number of observations N* from the *Basic* feature class were also utilized. The rest of features from classes described in Subsection 2.2 were excluded, mainly due to requiring additional evaluations of the objective function f.

Classification Tree for Covariance Functions The classification tree T depicted in Figure 1 has been obtained in [29] and represents the influence of landscape features on the most suitable covariance function. To train the tree T, all the sets described by features in the previous paragraph were divided into 8 classes according to which of the 8 considered GP model settings achieved the lowest RDE_{μ}. The tree was trained using the MATLAB implementation of the CART algorithm [4], where all features were considered as continuous variables. The fully-grown tree was pruned to depth 8 resulting in the shown tree T. The set of training points and the respective population is denoted $\mathcal{T}_{\mathcal{P}} = \mathcal{T} \cup \{(\mathbf{x}, \circ) | \, \forall \, \mathbf{x} \in \mathcal{P}\}$, where $\circ$ indicates the unknown fitness value of a point from the current population $\mathcal{P}$.

[1] Source code covering all mentioned experiments is available on `http://uivty.cs.cas.cz/~cma/ecml2019/source.zip`

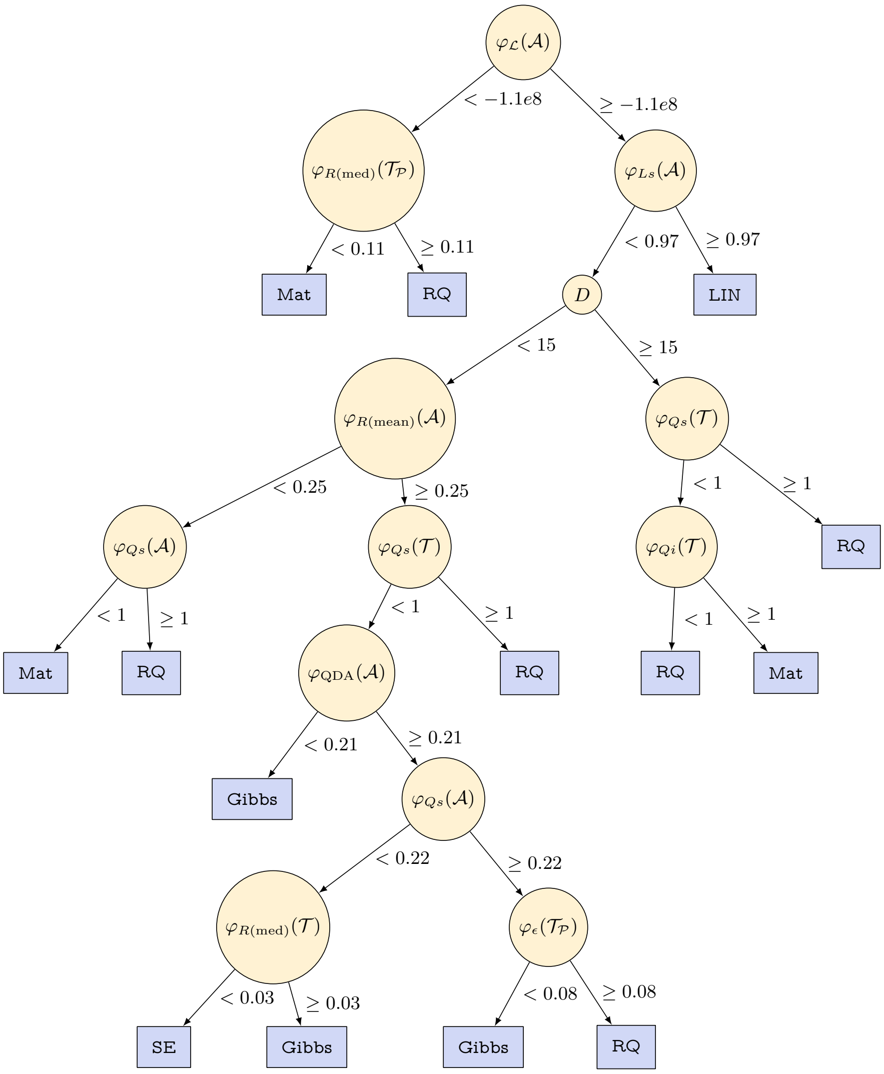

$\varphi_{\mathcal{L}}(\mathcal{A})$
$< -1.1e8$
$\geq -1.1e8$
$\varphi_{R(\mathrm{med})}(\mathcal{T}_{\mathcal{P}})$
$\varphi_{Ls}(\mathcal{A})$
< 0.11
≥ 0.11
< 0.97
≥ 0.97
Mat
RQ
D
LIN
< 15
≥ 15
$\varphi_{R(\mathrm{mean})}(\mathcal{A})$
$\varphi_{Qs}(\mathcal{T})$
< 0.25
≥ 0.25
< 1
≥ 1
$\varphi_{Qs}(\mathcal{A})$
$\varphi_{Qs}(\mathcal{T})$
$\varphi_{Qi}(\mathcal{T})$
RQ
< 1
≥ 1
< 1
≥ 1
< 1
≥ 1
Mat
RQ
$\varphi_{\mathrm{QDA}}(\mathcal{A})$
RQ
RQ
Mat
< 0.21
≥ 0.21
Gibbs
$\varphi_{Qs}(\mathcal{A})$
< 0.22
≥ 0.22
$\varphi_{R(\mathrm{med})}(\mathcal{T})$
$\varphi_{\epsilon}(\mathcal{T}_{\mathcal{P}})$
< 0.03
≥ 0.03
< 0.08
≥ 0.08
SE
Gibbs
Gibbs
RQ

Figure 1: Classification tree T selecting the most suitable covariance function based on landscape features [29]. In each iteration of the DTS-CMA-ES, the landscape features in the splitting nodes are calculated on sets in brackets, i. e., archive of points evaluated so far $\mathcal{A}$, GP model training set $\mathcal{T}$, and the set of training points and current population $\mathcal{T}_{\mathcal{P}} = \mathcal{T} \cup \{(\mathbf{x}, \circ) | \forall \mathbf{x} \in \mathcal{P}\}$, where $\circ$ indicates an unknown fitness value of a point from the current population $\mathcal{P}$. The covariance function is determined by the leaf reached by the sequence of splitting nodes decisions. The features used for node splits are explained in the text of Subsection 4.1. Covariances in leaves are listed in Table 1.

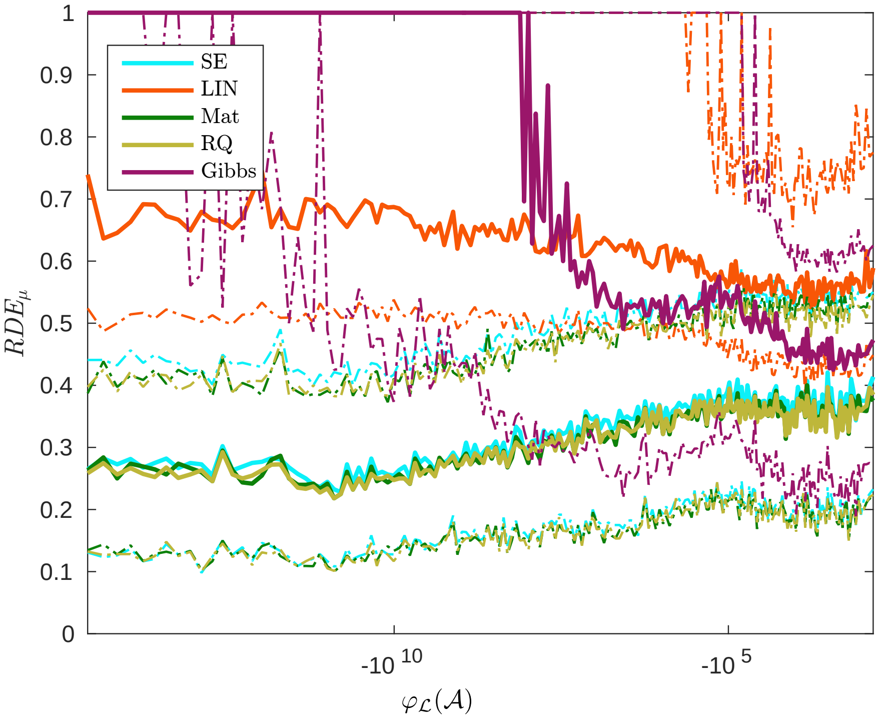


Figure 2: Median (solid lines) and 1st/3rd quartiles (dash-dot lines) of RDE_μ values dependency on $\varphi_{\mathcal{L}}(\mathcal{A})$ for all tested covariances calculated on all available datasets.

The features employed in the tree T represent various landscape properties: D is the dimension of the investigated function; $\varphi_{\mathcal{L}}$ is the log-likelihood of the set of points $\mathbf{X}$ with respect to the CMA-ES sampling distribution [29] (see Figure 2 for the average RDE_μ dependency on $\varphi_{\mathcal{L}}(\mathcal{A})$[2]); $\varphi_{R(\text{mean})}$ and $\varphi_{R(\text{med})}$ denote two ratios of the mean and median distances of the 'best' objectives vs. 'all' objectives [21]; φ_{Ls}, φ_{Qs}, and φ_{Qi} represent the adjusted R^2 (i. e., the model fit) of linear, quadratic simple, and quadratic with interactions fitted regression models [22]; φ_{QDA} is the mean missclassification error of Quadratic Discriminant Analysis on points divided into two classes according to the fitness values with median as a threshold [22]; φ_ϵ denotes the argument of the maximum information content of the fitness sequence [24].

The covariance functions located in leaves of the tree T are listed in Table 1.

[2] Figures of the RDE_μ dependencies on the remaining features can be found on an authors' webpage: `http://uivty.cs.cas.cz/~cma/ecml2019/`.

Table 1: Considered GP covariance functions. Notation: d – metric measuring the distance $d(\mathbf{x}_p, \mathbf{x}_q)$, hyperparameters σ_0 – scalar multiplication factor, σ_f^2 – signal variance, ℓ – characteristic lenght-scale (spatially varying in the Gibbs [9] covariance, where $\ell(\mathbf{x})$ is an arbitrary positive function of $\mathbf{x}$), and $\alpha > 0$.

name	kernel
linear (LIN)	$\mathbf{K}_{\mathrm{LIN}}(\mathbf{x}_p, \mathbf{x}_q) = \sigma_0^2 + \mathbf{x}_p^\top \mathbf{x}_q$
squared-exponential (SE)	$\mathbf{K}_{\mathrm{SE}}(d; \sigma_f, \ell) = \sigma_f^2 \exp\left(-\frac{d^2}{2\ell^2}\right)$
rational quadratic (RQ)	$\mathbf{K}_{\mathrm{RQ}}(d; \sigma_f, \ell) = \sigma_f^2 \left(1 + \frac{d^2}{2\ell^2\alpha}\right)^{-\alpha}$
Matérn $\frac{5}{2}$ [31] (Mat)	$\mathbf{K}_{\mathrm{Mat}}^{\frac{5}{2}}(d; \sigma_f, \ell) = \sigma_f^2 \left(1 + \frac{\sqrt{5}d}{\ell} + \frac{5d^2}{3\ell^2}\right) \exp\left(-\frac{\sqrt{5}d}{\ell}\right)$
Gibbs [9]	$\mathbf{K}_{\mathrm{Gibbs}}(\mathbf{x}_p, \mathbf{x}_q) = \sigma_f^2 \left(\frac{2\ell(\mathbf{x}_p)\ell(\mathbf{x}_q)}{\ell(\mathbf{x}_p)^2 + \ell(\mathbf{x}_q)^2}\right)^{D/2} \exp\left(-\frac{(\mathbf{x}_p - \mathbf{x}_q)^\top(\mathbf{x}_p - \mathbf{x}_q)}{\ell(\mathbf{x}_p)^2 + \ell(\mathbf{x}_q)^2}\right)$

Algorithm 1 Covariance function selection in DTS-CMA-ES model training

```
Input: $\mathcal{A}$ (archive), $\mathcal{P}$ (population), $N_{\max}$ (maximum training set size), $TSS$ (training
    set selection method), $r$ (maximal radius of selected points), $\mu$ (GP mean function),
    $\sigma$ (CMA-ES step-size), $\mathbf{C}$ (CMA-ES covariance function)
1: $\{(\mathbf{x}_k, y_k)\}_{k=1}^{N_{\max}} \leftarrow$ select max. $N_{\max}$ points from $\mathcal{A}$ using $TSS$ and $r$
2: $\mathbf{x}_k \leftarrow$ transform $\mathbf{x}_k$ into the $(\sigma)^2\mathbf{C}$ basis        $k = 1, \ldots, N_{\max}$
3: $y_k \leftarrow$ normalize $y_k$ to zero mean and unit variance        $k = 1, \ldots, N_{\max}$
4: $\mathbf{K} \leftarrow T(\mathcal{A}, \mathcal{T} = \{(\mathbf{x}_k, y_k)\}_{k=1}^{N_{\max}}, \mathcal{P})$
5: $\boldsymbol{\theta} \leftarrow$ fit the hyperparameters of $(\mu, \mathbf{K})$ by likelihood maximization
Output: $M$ – trained GP model with hyperparameters $\boldsymbol{\theta}$
```

4.2 Application phase

Covariance Function Selection The implementation of the selection of the covariance function for the DTS-CMA-ES based on the classification tree T is quite straightforward. We have modified the original algorithm only in the GP model training method (see Algorithm 1). We have incorporated an additional step applying covariance function selection using the classification tree T between the training set transformation and fitting the GP hyperparameters $\boldsymbol{\theta}$.

Covariance selection validation setup We have compared the described adaptive DTS-CMA-ES that online chooses the covariance function using the tree T (denoted as T-DTS) with five DTS-CMA-ES versions that use solely one covariance from Table 1. The comparison was performed on the noiseless part of the COCO framework using instances 1–5 and 81–90 of all 24 benchmark functions in dimensions 2, 3, 5, 10, and 20. Each of the six DTS-CMA-ES versions had a budget of $250D$ fitness function evaluations to reach the target value 10^{-8} from the function optimum. Except the choice of the covariance function, the DTS-CMA-ES was tested in its non-adaptive version using the overall best settings from [3].

4.3 Results

Results from the comparison of six DTS-CMA-ES versions are depicted in Table 2 and Figures 3 and 4. The graphs in Figures 3 and 4 show the dependence of the scaled best-achieved logarithms $\varDelta_f^{\log}$ of median distances $\varDelta_f^{\mathrm{med}}$ to the optimal fitness value on the number of cost-aware fitness evaluations divided by the dimension. Medians $\varDelta_f^{\mathrm{med}}$, 1st, and 3rd quartiles are calculated from 15 independent instances for each respective algorithm, function, and dimension. The scaled logarithms of $\varDelta_f^{\mathrm{med}}$ are calculated as

$$\varDelta_f^{\log} = \frac{\log \varDelta_f^{\mathrm{med}} - \varDelta_f^{\mathrm{MIN}}}{\varDelta_f^{\mathrm{MAX}} - \varDelta_f^{\mathrm{MIN}}} \log_{10}\left(1/10^{-8}\right) + \log_{10} 10^{-8}\,, \tag{2}$$

where $\varDelta_f^{\mathrm{MIN}}$ ($\varDelta_f^{\mathrm{MAX}}$) is the minimal (maximal) distance $\log \varDelta_f^{\mathrm{med}}$ found among all the compared algorithms for the particular function f and dimension D between 0 and 250 function evaluations per D. The resulting values are scaled to interval $[-8, 0]$, where -8 corresponds to $\varDelta_f^{\mathrm{MIN}}$ and 0 to $\varDelta_f^{\mathrm{MAX}}$. More detailed results can be found on an authors' webpage[3].

We have tested the statistical significance of performance differences on 24 COCO functions in $5D$ using the Iman and Davenport's improvement of the Friedman test [6]. The test was conducted separately for two function evaluation budgets. Let $\#\mathrm{FE}_\mathrm{T}$ be the smallest number of function evaluations at which at least one DTS-CMA-ES version reached the precision $\varDelta_f^{\mathrm{med}} \leq 10^{-8}$, or $\#\mathrm{FE}_\mathrm{T} = 250D$ if no version reached the precision within $250D$ evaluations. The DTS-CMA-ES versions are ranked on each COCO function with respect to $\varDelta_f^{\mathrm{med}}$ at a given budget of function evaluations. The null hypothesis of equal performance of all versions is rejected for the higher function evaluation budget $\#\mathrm{FEs} = \#\mathrm{FE}_\mathrm{T}$, as well as for the lower budget $\#\mathrm{FEs} = \frac{\#\mathrm{FE}_\mathrm{T}}{4}$ (in both cases, $p < 10^{-3}$).

We test pairwise differences in the performance using the post-hoc Friedman test [8] with the Bergmann-Hommel correction controlling the family-wise error. The numbers of functions at which one DTS-CMA-ES version achieved a higher rank than the other are enlisted in Table 2. The table also contains the pairwise statistical significances.

From the results in Table 2 and in Figures 3 and 4, we can consider the results of the T-DTS, and the DTS-CMA-ES with SE, Mat, and RQ covariances being statistically equivalent meaning that neither of them is significantly better than the other one. Looking on the detailed results on the authors' webpage[3], those covariances provided the best performance on the functions f_5, f_{8-11}, and f_{14}. On the other hand, slightly worse results can be observed on functions f_7, f_{13}, f_{16}, and f_{20}. On functions f_6 and $f_{17,18}$ the T-DTS results more or less follow SE, Mat, and RQ performance although the best performance was provided by the Gibbs covariance. The results on multimodal functions f_{22-24} show increasing T-DTS performance with growing dimension. The versions using LIN and Gibbs

[3] http://uivty.cs.cas.cz/~cma/ecml2019/

Table 2: A pairwise comparison of the algorithms in $5D$ over the COCO for different evaluation budgets. The number of wins of the i-th algorithm against the j-th algorithm over all benchmark functions is given in i-th row and j-th column. The asterisk marks the row algorithm being significantly better than the column algorithm according to the Friedman post-hoc test with the Bergmann-Hommel correction at the family-wise significance level $\alpha = 0.05$.

$\mathbf{5D}$	T-DTS		LIN		SE		Matérn		RQ		Gibbs	
$^{\#\text{FEs}}/_{\#\text{FE}_\text{T}}$	$^1/_4$	1	$^1/_4$	1	$^1/_4$	1	$^1/_4$	1	$^1/_4$	1	$^1/_4$	1
T-DTS	—	—	22.5^*	24^*	10.5	12	11.5	12	12.5	11	17.5	22.5^*
LIN	1.5	0	—	—	0.5	0	0.5	0	0.5	0	0.5	0
SE	13.5	12	23.5^*	24^*	—	—	10.5	11.5	13.5	9.5	15.5	20^*
Matérn	12.5	12	23.5^*	24^*	13.5	12.5	—	—	10.5	10.5	16.5	23.5^*
RQ	11.5	13	23.5^*	24^*	10.5	14.5	13.5	13.5	—	—	14.5	22^*
Gibbs	6.5	1.5	23.5^*	24	8.5	4	7.5	0.5	9.5	2	—	—

covariance functions provide considerably lower performance in comparison with the remainder. Variability of length-scale utilized by Gibbs covariance function helps the DTS-CMA-ES to converge on hard-to-regress f_6 and on multimodal *Schaffer*'s functions $f_{17,18}$ especially in higher dimensions, where the performance of DTS-CMA-ES using Gibbs covariance in GP model is the best of all compared versions.

A possible reason of the T-DTS results may lie in an imbalance of the input dataset for decision tree. Covariances SE, Mat, and RQ performed almost similar and, in average, provided the overall best prediction performance among tested covariances on the set of datasets $\mathcal{D}$. Therefore, these three covariances were marked as best on most of datasets and the remaining two (LIN and Gibbs) were best on minority of datasets. The trained classification tree was probably not able to capture such imbalance of the input data and predicted LIN or Gibbs as the most convenient covariances more often than it was necessary.

5 Conclusion and Future work

This article investigates the surrogate model selection problem for continuous single-objective black-box optimizers in the context of reusing knowledge through landscape analysis. The proposed concept was applied to select a hyperparameter of Gaussian process models, namely the covariance function, and was utilized during the DTS-CMA-ES run to save costly fitness evaluations. The DTS-CMA-ES upgraded with hyperparameter selection was compared to five DTS-CMA-ES versions using different covariances on the set of noiseless benchmarks.

The presented proof of concept has shown that the methodology can be utilized for hyperparameter selection. The tree-assisted DTS-CMA-ES had a performance equivalent to DTS-CMA-ES versions with successful fixed covariance functions. On the other hand, the classification tree as a mapping of values of

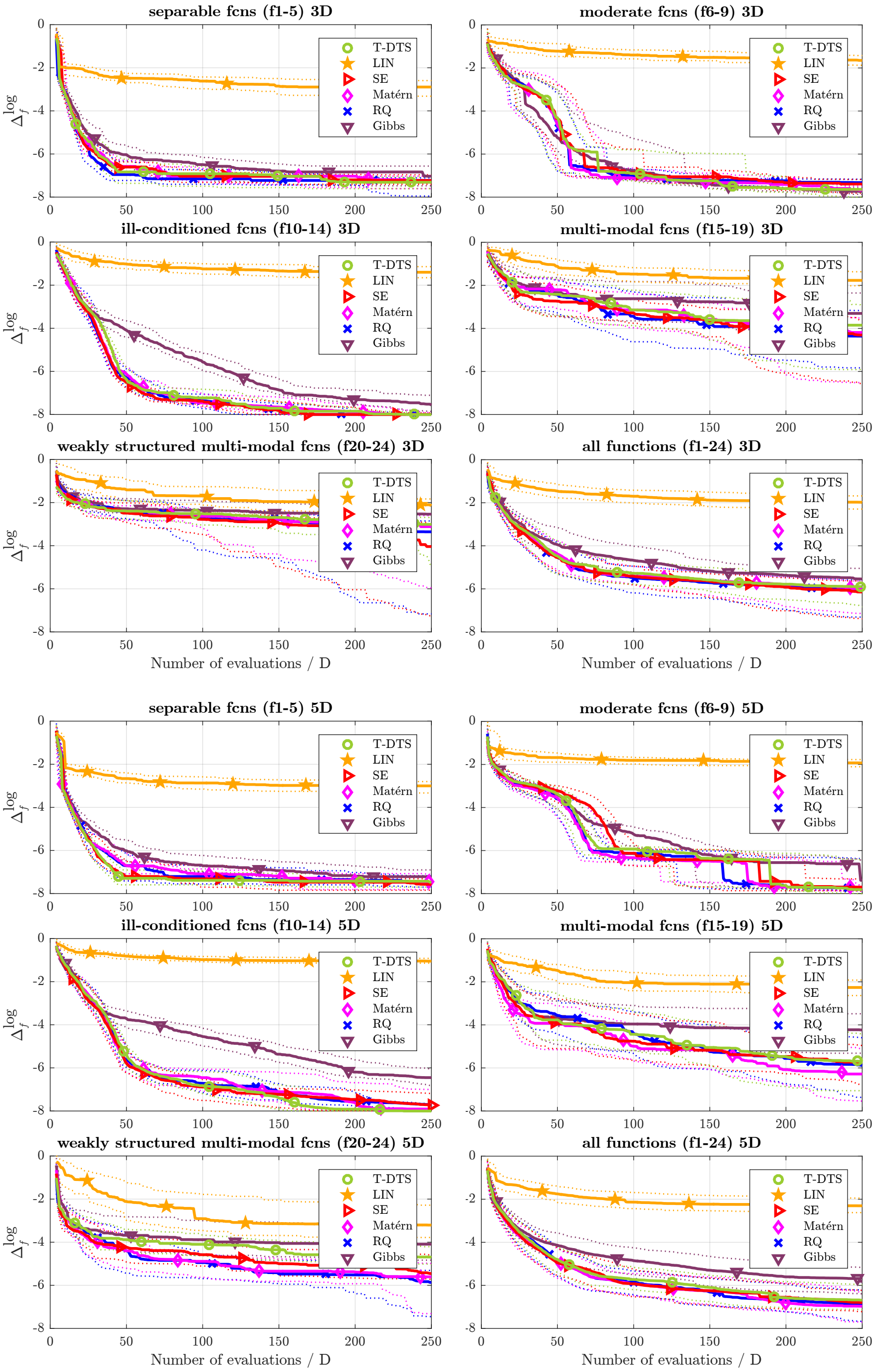


Figure 3: Scaled medians (solid) and 1st/3rd quartiles (dotted) distances $\Delta_f^{\log}$ averaged over the groups of noiseless COCO functions in $3D$ and $5D$ for different settings of DTS-CMA-ES GP covariance function.

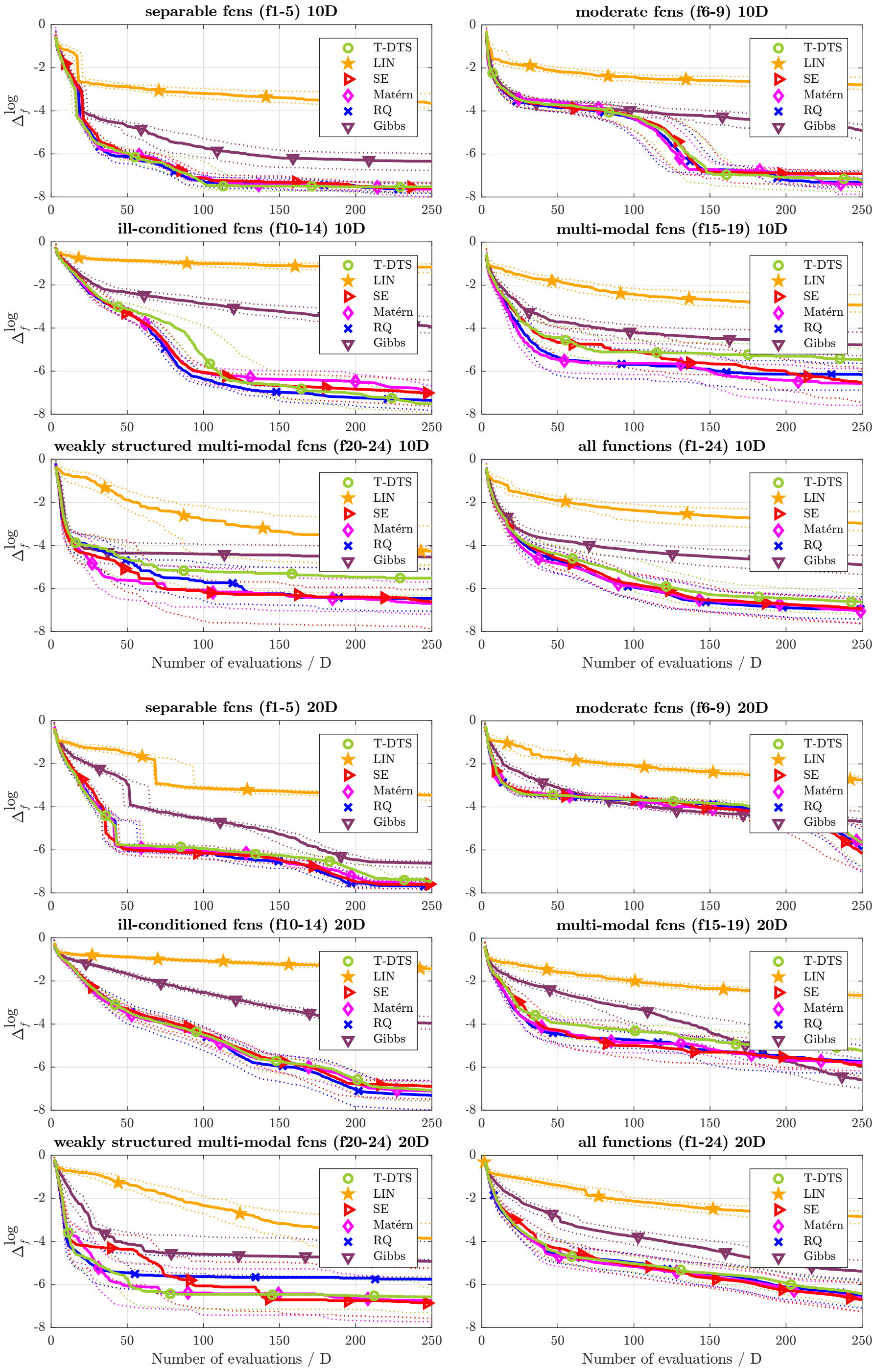


Figure 4: Scaled medians (solid) and $1^{\text{st}}/3^{\text{rd}}$ quartiles (dotted) distances $\Delta_f^{\log}$ averaged over the groups of noiseless COCO functions in $10D$ and $20D$ for different settings of DTS-CMA-ES GP covariance function.

landscape features to the covariance functions for the DTS-CMA-ES seems not to have learned very accurately.

Future research should be focused mostly on deeper understanding of the surrogate model selection problem and the possibilities of landscape analysis in this context. The investigation of various mappings to models and their hyperparameters capable to capture relationships between landscape features and surrogate model performance is definitely needed. Another direction is to extend the presented research also to other kinds of surrogate models.

Acknowledgements The reported research was supported by the Czech Science Foundation grants Nos. 17-01251S and 18-18080S and by the Grant Agency of the Czech Technical University in Prague with its grant No. SGS17/193/OHK4/3T/14. Further, access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

References

1. Auger, A., Schoenauer, M., Vanhaecke, N.: LS-CMA-ES: A second-order algorithm for covariance matrix adaptation. In: Parallel Problem Solving from Nature - PPSN VIII. pp. 182–191 (2004)
2. Bajer, L., Pitra, Z., Holeňa, M.: Benchmarking Gaussian processes and random forests surrogate models on the BBOB noiseless testbed. In: Proceedings of the 17th GECCO Conference Companion. ACM, New York, Madrid (July 2015)
3. Bajer, L., Pitra, Z., Repický, J., Holeňa, M.: Gaussian process surrogate models for the CMA Evolution Strategy. Evolutionary Computation **0**(0), 1–33 (0). https://doi.org/10.1162/evco_a_00244, pMID: 30540493
4. Breiman, L.: Classification and regression trees. Chapman & Hall/CRC (1984)
5. Büche, D., Schraudolph, N.N., Koumoutsakos, P.: Accelerating evolutionary algorithms with Gaussian process fitness function models. IEEE Transactions on Systems, Man, and Cybernetics, Part C **35**(2), 183–194 (2005)
6. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**, 1–30 (2006)
7. Flamm, C., Hofacker, I.L., Stadler, P.F., Wolfinger, M.T.: Barrier Trees of Degenerate Landscapes. Zeitschrift für Physikalische Chemie International Journal of Research in Physical Chemistry and Chemical Physics **216**(2), 155–173 (2002)
8. García, S., Herrera, F.: An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. Journal of Machine Learning Research **9**, 2677–2694 (2008)
9. Gibbs, M.N.: Bayesian Gaussian Processes for Regression and Classification. Ph.D. thesis, Department of Physics, University of Cambridge (1997)
10. Hansen, N.: The CMA evolution strategy: A comparing review. In: Towards a New Evolutionary Computation, pp. 75–102. No. 192 in Studies in Fuzziness and Soft Computing, Springer Berlin Heidelberg (Jan 2006)
11. Hansen, N., Auger, A., Finck, S., Ros, R.: Real-parameter black-box optimization benchmarking 2012: Experimental setup. Tech. rep., INRIA (2012)

12. Hansen, N., Finck, S., Ros, R., Auger, A.: Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Tech. Rep. RR-6829, INRIA (2009), updated February 2010
13. Hansen, N.: A Global Surrogate Assisted CMA-ES. In: GECCO. Prague, Czech Republic (Jul 2019). https://doi.org/10.1145/3321707.3321842
14. Jin, R., Chen, W., Simpson, T.: Comparative studies of metamodelling techniques under multiple modelling criteria. Structural and Multidisciplinary Optimization **23**(1), 1–13 (2001)
15. Kern, S., Hansen, N., Koumoutsakos, P.: Local Meta-models for Optimization Using Evolution Strategies. In: Parallel Problem Solving from Nature - PPSN IX. Lecture Notes in Computer Science, vol. 4193, pp. 939–948. Springer Berlin Heidelberg (2006)
16. Kerschke, P.: Comprehensive feature-based landscape analysis of continuous and constrained optimization problems using the R-package flacco. ArXiv e-prints (2017)
17. Kerschke, P., Preuss, M., Wessing, S., Trautmann, H.: Detecting funnel structures by means of exploratory landscape analysis. pp. 265–272. GECCO '15, ACM (2015)
18. Kerschke, P., Preuss, M., Hernández, C., Schütze, O., Sun, J.Q., Grimme, C., Rudolph, G., Bischl, B., Trautmann, H.: Cell mapping techniques for exploratory landscape analysis. In: EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V. pp. 115–131. Springer International Publishing (2014)
19. Lemke, C., Budka, M., Gabrys, B.: Metalearning: a survey of trends and technologies. Artificial Intelligence Review **44**(1), 117–130 (Jun 2015)
20. Loshchilov, I., Schoenauer, M., Sebag, M.: Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy. In: Proceedings of the 14th GECCO. pp. 321–328. GECCO '12, ACM, New York, NY, USA (2012)
21. Lunacek, M., Whitley, D.: The dispersion metric and the cma evolution strategy. pp. 477–484. GECCO '06, ACM (2006)
22. Mersmann, O., Bischl, B., Trautmann, H., Preuss, M., Weihs, C., Rudolph, G.: Exploratory landscape analysis. pp. 829–836. GECCO '11, ACM (2011)
23. Mersmann, O., Preuss, M., Trautmann, H.: Benchmarking evolutionary algorithms: Towards exploratory landscape analysis. pp. 73–82. PPSN XI, Springer Berlin Heidelberg (2010)
24. Muñoz, M.A., Kirley, M., Halgamuge, S.K.: Exploratory landscape analysis of continuous space optimization problems using information content. IEEE Transactions on Evolutionary Computation **19**(1), 74–87 (2015)
25. Muñoz, M.A., Sun, Y., Kirley, M., Halgamuge, S.K.: Algorithm selection for black-box continuous optimization problems. Inf. Sci. **317**(C), 224–245 (2015)
26. Myers, R., Montgomery, D.: Response Surface Methodology: Process and Product in Optimization Using Designed Experiments. John Wiley & Sons, Inc., New York, NY, USA, 1st edn. (1995)
27. Pitra, Z., Bajer, L., Holeňa, M.: Doubly trained evolution control for the Surrogate CMA-ES. In: Proceedings of the PPSN XIV: 14th International Conference, Edinburgh, UK, September 17-21. pp. 59–68. Springer International Publishing, Cham (2016)
28. Pitra, Z., Bajer, L., Repický, J., Holeňa, M.: Overview of surrogate-model versions of Covariance Matrix Adaptation Evolution Strategy. GECCO '17, ACM (2017)
29. Pitra, Z., Repický, J., Holeňa, M.: Landscape analysis of Gaussian process surrogates for the covariance matrix adaptation evolution strategy. pp. 691–699. GECCO '19, ACM (2019)

30. Pitra, Z., Bajer, L., Repický, J., Holeňa, M.: Transfer of knowledge for surrogate model selection in cost-aware optimization. In: Krempl, G., Lemaire, V., Kottke, D., Calma, A., Holzinger, A., Polikar, R., Sick, B. (eds.) ECML PKDD 2018: Workshop on Interactive Adaptive Learning. Proceedings. pp. 89–94. ECML PKDD 2018, Dublin, Ireland (Sep 2018)
31. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. Adaptative computation and machine learning series, MIT Press (2006)
32. Repický, J., Holeňa, M.: Automated selection of covariance function for Gaussian process surrogate models. In: ITAT 2018 Proceedings. CEUR Workshop Proceedings, vol. 2203, pp. 64–71. CEUR-WS.org (2018)
33. Rice, J.R.: The algorithm selection problem. Advances in Computers, vol. 15, pp. 65 – 118. Elsevier (1976)
34. Sun, C., Jin, Y., Cheng, R., Ding, J., Zeng, J.: Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems. IEEE Transactions on Evolutionary Computation (2017)
35. Ulmer, H., Streichert, F., Zell, A.: Evolution strategies assisted by Gaussian processes with improved preselection criterion. In: The 2003 Congress on Evolutionary Computation, 2003. CEC '03. vol. 1, pp. 692–699 Vol.1 (Dec 2003)
36. Yu, H., Tan, Y., Sun, C., Zeng, J., Jin, Y.: An adaptive model selection strategy for surrogate-assisted particle swarm optimization algorithm. pp. 1–8. SSCI '16 (2016)

Explicit Control of Feature Relevance and Selection Stability Through Pareto Optimality

Victor Hamer[1]* and Pierre Dupont[2]

UCLouvain - ICTEAM/INGI/Machine Learning Group, Place Sainte-Barbe 2,
B-1348 Louvain-la-Neuve, Belgium.
[1]victor.hamer@uclouvain.be [2]pierre.dupont@uclouvain.be

Abstract. Feature selection is an important issue when one deals with large amounts of *omics* data. Feature selection offers interpretability of the predictive model to the domain expert. Such interpretability is strongly affected by the typical instability of current feature selection methods. Instability here refers to the fact that the selected features may be drastically different even after marginal modification of the data. In this paper, we investigate the possibility of a tradeoff between the classification performance and the stability of a standard feature selection method: the Recursive Feature Elimination algorithm (RFE). The compromise is done by explicitly favoring the selection of some features through differential shrinkage. Such an approach allows the domain expert to control **explicitly** the trade-off between selection stability and predictive accuracy. Domain experts can thus select particular Pareto-optimal compromises, based on their personal preferences. As a secondary contribution, we propose the use of the hypervolume metric to assess the performance of methods realizing such a compromise and we define a corresponding confidence interval. Our approach is evaluated on prostate cancer diagnosis from microarray data and handwritten digit recognition tasks. Results show that the aforementioned tradeoff is effectively possible and that prior knowledge is an efficient way of stabilizing the selection.

Keywords: Feature selection · Selection stability · Classification performance · Transfer learning · Bi-objective optimization · Multitask learning

1 Introduction

Feature selection, *i.e.* the selection of a small subset of informative and relevant features to be included in a predictive model, has become compulsory for a wide variety of applications due to the appearance of very high dimensional datasets, notably in the biomedical data domain [20]. Filtering noisy and irrelevant features can avoid overfitting the data and potentially improve predictive performance. Feature selection also allows for the learning of fast and compact

[*] Corresponding author

models, which are easier to interpret. Such models can then be analyzed by domain experts and are easier to validate. Getting more interpretable models is also a key concern nowadays and even considered by many as a requirement when deployed in the medical domain.

Feature selection has been already largely studied. Yet, current methods are still widely unsatisfactory mainly because of the typical instability they exhibit. Instability here refers to the fact that the selected features may be drastically different for similar data, even though the true underlying processes (explaining the target variable) are essentially constant. Such instability is a key issue as it reduces the interpretability of the predictive models as well as the trust of domain experts towards the selected feature subsets. We address this problem here by designing methods balancing between the classification performance and the selection stability of the well-known Recursive Feature Elimination (RFE) algorithm. Our approach allows domain experts to explicitly control the trade-off and to select Pareto-optimal compromises based on their personal preferences.

In the rest of this section, two distinct stability problems that are tackled in this paper are introduced.

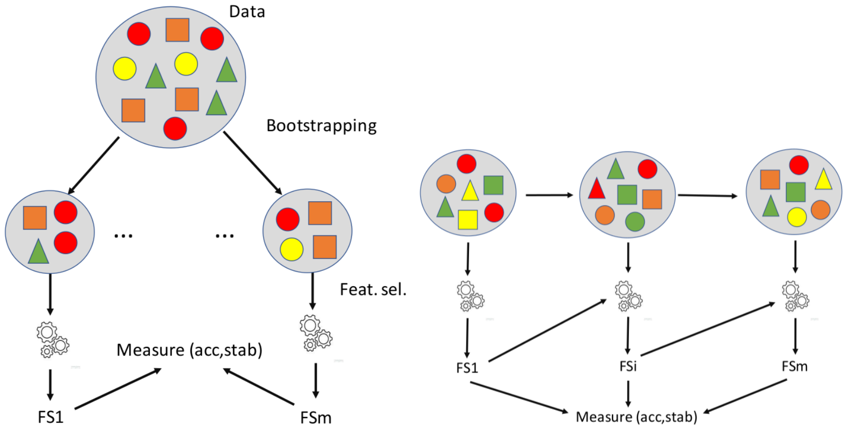
1.1 The Stability Problems

Single Task Stability (1) Feature selection methods are often inherently unstable, *i.e.* they return highly different feature sets when the training data is slightly modified. Figure 1a illustrates such an instability. The initial dataset is perturbed[1] to form different datasets. Instability arises when little overlap of the selected features occurs. This prevents a correct and sound interpretation of the selected features and strongly impacts their further validation by domain experts. Unlike optimizing the accuracy of predictive models, optimizing selection stability may look trivial since an algorithm always returning an arbitrary but fixed set of features would be stable by design. Yet, such an algorithm is not expected to select informative and predictive features. This illustrates that optimizing stability is only well-posed jointly with predictive accuracy, and possibly additional criteria such as minimal model size or sparsity.

Transfer Learning Selection Stability (2) Multi-task feature selection aims at discovering variables that are relevant for several similar, yet distinct, classification tasks. Different feature subsets can be returned for each task. In this paper, we focus on the case where all learning tasks are not directly available. Information from the tasks arising first can be propagated to subsequent tasks, via *transfer learning.* Stability has to be encouraged from the domain expert point of view as features that are relevant for different data sources are likely to be particularly interesting to study. The accuracy-stability trade-off on such a learning problem (represented in Figure 1b) can take two extreme values. With complete disregard to stability, each feature set could be selected on a given task

[1] Here by bootstrapping which is often used to measure such instability, but it could be any small perturbation.

independently of the others, with no control on the across task stability. On the contrary, maximum stability can trivially be reached by returning the feature set computed for the first task, for all subsequent tasks. However, this is expected to reduce the accuracy of the models built on these subsequent tasks as previously learned features might turn out to be less informative for them. This would be the case if the different tasks are obtained by gradually enriching or correcting the data as features learned on the error-corrected data are expected to be more relevant.



(a) Single-task **(b)** Transfer learning

Fig. 1. Illustration of two stability problems. For both problems, the outcome is a measure of the trade-off between prediction accuracy and selection stability. Methods allowing domain experts to control this trade-off are proposed in the subsequent parts of this paper.

In section 2, feature selection methods and propositions to increase stability are reviewed. Section 3 introduces a metric to assess the performance of methods compromising between feature selection stability and classification performance. Then a biased variant of the RFE algorithm is proposed in section 4. Section 5 demonstrates the ability of this biased RFE to tackle the previously mentioned stability problems.

2 Related Work

Feature selection techniques are generally split into three categories: filters, wrappers and embedded methods. *Filters* evaluate the relevance of features independently of the final model, most commonly a classifier, and remove low ranked features. Simple filters (*e.g.* t-test or ANOVA) are univariate, which is computationally efficient and tends to produce a relatively stable selection but they plainly ignore the possible dependencies between various features. Information theoretic methods, such as MRMR [7] and many others, are based on mutual information between features or with the response, but a robust estimation of these quantities in high dimensional spaces remains difficult. *Wrappers* look for the feature subset that will yield the best predictive performance on a validation set. They are classifier dependent and very often multivariate. However, they can be very computationally intensive and an optimal feature subset can rarely be found. *Embedded methods* select features by determining which features are more important in the decisions of a predictive model. Prominent examples include SVM-RFE [10] and logistic regression with a LASSO [24] or Elastic Net penalty [30]. These methods tend to be more computationally demanding than filters but they integrate into a single procedure the feature selection and the estimation of a predictive model. Yet, they also tend to produce much less stable models.

Some works specifically study the causes of selection instability. Results show that it is mostly caused by the small sample/feature ratio [2], noise in the data or imbalanced target variable [5] and feature redundancy [23]. While all of these reasons clearly play a role, the first one is likely the most important one in a biomedical domain with typically several thousands, if not millions, of features for only a few dozens or hundreds of samples. This is likely why stable feature selection is intrinsically hard in this domain and why existing techniques are still largely unsatisfactory.

Looking for a stable feature selection also requires a proper way to quantify stability itself and lots of measures have already been proposed: the Kuncheva index [15], the Jaccard index [14], the POG [21] and nPOG [26] indices among others. Under such a profusion of different measures, it becomes difficult to justify the choice of a particular index and even more to compare results of works based on different metrics. Furthermore, the large number of available measures can lead to publication bias (researchers may select the index that makes their algorithm look the most stable) [6]. In the hope of fixing this issue, a recent work [17] lists and analyzes 15 different stability measures. They are compared based on the satisfaction of 5 different properties that a stability measure should comply. A novel and unifying index has been proposed in this regard. This index, used throughout this paper, measures the stability across M selected subsets of features. It can be computed according to equation (1).

$$\phi = 1 - \frac{\frac{1}{d}\sum_{f=1}^{d} s_f^2}{\frac{k}{d} * (1 - \frac{k}{d})} \tag{1}$$

with $s_f^2 = \frac{M}{M-1}\hat{p}_f(1-\hat{p}_f)$ the estimator of the variance of the selection of the f_{th} feature over the M selected subsets and k the mean number of features selected from the original d features.[2] This measure is the only existing measure satisfying the 5 (good) properties described in [17], namely *fully defined*, *strict monotonicity*, *bounds*, *maximum stability* $\Leftrightarrow$ *deterministic selection* and *correc-*

[2] $\hat{p}_f$ is the fraction of times feature f has been selected among the M subsets.

tion for chance. It is formally bounded by -1 and 1 but is asymptotically lower bounded by 0 as $M \to \infty$. It is also equivalent to the Kuncheva Index (KI)[15] when the number of selected features k is constant across the M selected subsets but can be computed in $O(M * d)$ time, whereas KI can only be computed in $O(M^2 * d)$.

Several authors already proposed different methods to increase stability. For instance, instance-weighting for variance reduction [11] which tends to increase feature stability while keeping a comparable predictive performance. Ensemble methods for feature selection have also been proposed [1] and generally increase feature stability. Nonetheless, the gain in stability offered by existing methods is still limited and, maybe more importantly, the stability of the selection cannot be controlled explicitly, which is the main goal of this paper.

Multi-task feature selection has already been largely studied [27]. Encouraging the selection of common predictors across tasks can be done by using the ℓ_1/ℓ_p regularization scheme. The cost of selecting different predictors for different tasks can be controlled by using different norms ℓ_1/ℓ_p, as $p \to \infty$ favors the selection of common features. As with the differential shrinkage approach proposed here, penalties caused by selecting several times the same feature are reduced. Notably, the ℓ_1/ℓ_∞ [16] and ℓ_1/ℓ_2 [18,19] penalties have been studied in details. Efficient projected gradient algorithms, for general p, are proposed and the effect of p on the shared sparsity pattern and the classification performance is analyzed [25]. The main goal of [25] is to find adequate feature-sharing degrees such as to maximize the prediction performance of the models, which is different from the objective of explicit control of the accuracy-stability trade-off that is pursued in the present paper. Although this approach has been originally introduced for standard multi-task feature selection, it can trivially be adapted to the transfer learning setting [25]. Other similar approaches have also been proposed [3,4,8] (see [27] for a complete survey).

3 A Multi-Objective Evaluation Framework Through Pareto Optimality

In this section, we propose to use a classical evaluation framework in multi-objective optimization to assess the efficiency of methods balancing between classification performance and selection stability. An (accuracy,stability) pair[3] (a_1, ϕ_1) dominates another pair (a_2, ϕ_2) iff $a_1 \geq a_2 \wedge \phi_1 \geq \phi_2$ and at least one of the inequalities is strict ($>$). A given method m is able to generate some pairs P_m in the space of all possible pairs[4] $P = \{(a, \phi) : 0 \leq a, \phi \leq 1\}$. From the set of generated pairs P_m, the set of pairs that are not dominated by any other pair,

[3] Common alternatives to the classification accuracy, such as specificity/sensitivity or AUC, can also be used.

[4] The careful reader may remember that the stability measure ϕ formally lies in the $[-1, 1]$ interval. However, as $\phi = 0$ corresponds to the stability of a uniformly random selection, we argue that the only interesting part of the stability spectrum is in fact $[0, 1]$.

Pa_m, can be found. This set, called the Pareto set, defines a subspace where no point dominates any other point. A domain expert would then choose his favorite pair based on his personal preference towards classification performance and feature selection stability.

As performance metric, we propose the widely used hypervolume measure [29], also known as S-metric. This volume represents the space containing the sets of accuracy-stability pairs that are dominated by at least one point of the Pareto set Pa_m. The hypervolume measure has the convenient property that whenever a Pareto set dominates another, the hypervolume of the former is greater. As our objective space is bidimensional, the hypervolume measure is referred to as the Dominance Area (DA) in the rest of this work.

An example of the DA metric can be seen in Figure 2. The blue method starts from the left with a higher accuracy. It thus gains some area over the red method. Nonetheless, the red method can reach higher stabilities without dropping the accuracy as much as the blue one. Overall, the red method has a larger DA. Note that this DA is also equal to the fraction of the total area that is dominated by the method, or 1 minus the fraction of area that dominates the method. Its value thus lies in the $[0, 1]$ interval.

As noticed by [28], this DA measure is biased towards convex, inner parts of the objective space. [28] tackles this problem by giving different weights to different portions of the objective space. This weighted DA can be computed via the weighted integral

$$DA_P = \int_0^1 \int_0^1 w(a,\phi) f_P(a,\phi) da d\phi \tag{2}$$

with w the weighting function and f_P the attainment function which is equal to 1 if (a, ϕ) is dominated and 0 otherwise. To preserve the $[0-1]$ bounds, w has to be normalized such that its integral over the objective space is 1. For example, the normalized weighting function $w_a(a,\phi) \triangleq \frac{e^{A*a}}{e^A-1}$ gives a higher weight to the portions of the space where the accuracy is high. In the example of Figure 2, the blue method actually outperforms the red one for $A > 2.5$. For the sake of generality, our methods are evaluated with $w(a, \phi) = 1$ but the proposed evaluation framework allows for more, for instance, if domain experts are particularly interested in some parts of the objective space.

In order to evaluate the pair $(a, \phi) \in P$ corresponding, for instance, to a set of meta-parameters, some data has to be used to learn the features and some data to evaluate them on independent examples. This can be done via standard cross-validation or by bootstrapping. Each set of meta-parameters produces different pairs in P and their average value is reported. Concretely, each point in Figure 2 comes with an uncertainty linked to the sampling of the data. In the following, we define a confidence interval on the true value of DA based on the derivation of confidence *regions* for each Pareto-optimal pair.

Let A be the random variable representing the accuracy value measured on a given subsampling of the data and Φ be the corresponding stability value. Let $\mathcal{P} = (A, \Phi)$ be the multivariate random variable with the accuracy and

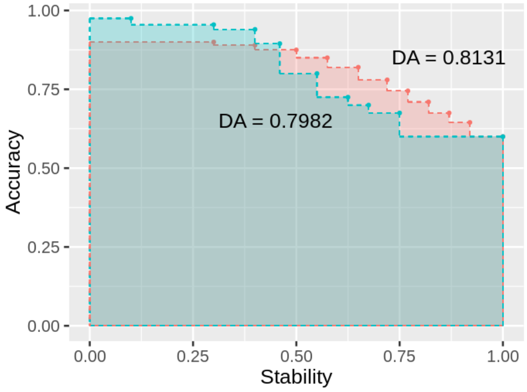


Fig. 2. Dominace Area (DA) toy example. The DA metric represents the area that is dominated by at least one point generated by the method.

stability as dimensions. Let us assume that the evaluation protocol produces B measurements of $\mathcal{P}$ for each Pareto-optimal point[5], represented by the vector $\mathbf{p}$. The Hotelling distribution T^2 is the multivariate counterpart of the Student's t distribution, with which we can define confidence (here 2-dimensional) regions.

$$T^2 = B(\bar{\mathbf{p}} - \mu(\mathbf{p}))'C^{-1}(\bar{\mathbf{p}} - \mu(\mathbf{p})) \sim \frac{2(B-1)}{B-2} * \mathcal{F}_{2,B-2} \tag{3}$$

with C the sample covariance matrix. It can be shown that T^2 is distributed like a Fisher distribution $\mathcal{F}_{2,B-2}$. Thus,

$$P\left[(\bar{\mathbf{p}} - \mu(\mathbf{p}))'C^{-1}(\bar{\mathbf{p}} - \mu(\mathbf{p}) \leq \frac{2*(B-1)}{B-2} * \mathcal{F}_{2,B-2}(\alpha)\right] = 1 - \alpha \tag{4}$$

The inequality defines an ellipsoidal region, that is likely to cover $\mu(\mathbf{p})$. The center of the ellipsoid is $\bar{\mathbf{p}}$. The length of the axis and their angle can be found by computing the eigenvalues and eigenvectors of the sample covariance matrix C. To compute a confidence interval on the DA, the most dominant and dominated point of each ellipse are found and used to compute the upper and lower bound of the confidence interval (see Figure 3c and 3d for concrete examples in our experiments).

4 A Biased Variant of the RFE Algorithm

In this section, we propose a simple method to balance between the classification performance and selection stability of a logistic RFE algorithm. The RFE

[5] B could be *e.g.* the number of bootstrap samples.

algorithm was originally introduced with a hinge loss. We prefer here the logistic variant for an expected smoother control of the trade-off under study. RFE iteratively drops the least relevant features until the desired number of features k is reached. We opt here to drop a fixed fraction (20%) of the features at each iteration. The loss function that a logistic RFE minimizes for a binary classification task is the following, with n the number of samples, $\mathbf{x_i}$ sample number i made of d features as dimensions, and y_i its label.

$$L = \sum_{i=1}^{n} \log(1 + e^{-y_i * (\mathbf{w} * \mathbf{x_i})}) + \lambda ||\mathbf{w}||_2 \tag{5}$$

The weight vector $\mathbf{w}$ contains a weight assigned to each feature. The features are then ranked based on the absolute value of their weight, which represents the importance of the feature in the final prediction. The term $\lambda||\mathbf{w}||_2$ of the loss function is a regularization term, preventing coefficients of the model to take too high values, which would most likely result in overfitting. In the classical approach, every feature is regularized by the same amount λ.

We propose to extend equation (5), such that the regularization term becomes $\lambda\boldsymbol{\beta}||\mathbf{w}||_2$. The function of the vector $\boldsymbol{\beta}$ is to bias the selection towards certain features via differential shrinkage. A feature f with a small β_f is less regularized and vice-versa. Its selection in the model is less penalized than a feature with a higher β_f. The search is thus *biased* towards features with small β_f. A similar differential shrinkage has already been applied to the ℓ_1-AROM and ℓ_2-AROM methods [12,13] with the objective of biasing the selection towards *a priori* relevant features or in a transfer learning context. In the remaining part of this section, three possible schemes to set the $\boldsymbol{\beta}$ vector, according to the setting of interest, are discussed.

Biased RFE for Single Task Feature Selection By varying the distribution of $\boldsymbol{\beta}$, the accuracy-stability trade-off of the biased RFE can be controlled. The biased RFE is equivalent to a standard RFE when $\boldsymbol{\beta} = \mathbf{1}$. Otherwise, the selection is biased towards features with a small β_f. This is expected to increase stability at the possible cost of some classification performance, as uninformative features could be prioritized. In this initial approach, we decide to favor some features non-uniformly at random, following a gamma distribution.

$$\beta_f \sim \mathit{\Gamma}(\alpha, 1)$$

with α the shape of the gamma distribution, which controls the trade-off. All β_f are post centered such that $\mu(\boldsymbol{\beta}) = 1$. As $\alpha \to \infty$, the gamma distribution tends to a Dirac delta, $\delta(\alpha)$. All features have then the same weight (equal to $\mu(\boldsymbol{\beta}) = 1$) and no bias is put in the selection. As $\alpha \to 0$, the distribution of $\boldsymbol{\beta}$ departs from $\delta(\alpha)$ which increases the bias. Domains experts can thus play with the α values and therefore explicitly tune the trade-off between selection stability and prediction accuracy.

Using Prior Knowledge The biased RFE can take advantage of available prior knowledge. If a ranking of the features is known, then the β_f can be assigned such that this ranking is respected. If the prior knowledge is meaningfull, the selection is no longer biased towards arbitrary features, but towards features that are high in the ranking, and thus potentially informative. Another type of prior knowledge could be an unordered set of features that are suspected to take part in the process of interest. The lowest β_f could then systematically be assigned to those features.

Biased RFE for Transfer Learning We are now interested in the across task stability that can be obtained via transfer learning. Tasks are thus ranked such that information from previous tasks can be used in the selection of features for subsequent tasks.[6] In task i, features that have been returned for tasks $0..i-1$ should be prioritized over the rest, such that the feature stability is increased. Given the definition of stability used here (equation (1)), it is actually possible to compute the drop/gain in stability that the selection of a feature would cause. Intuitively, we propose to bias the selection, through a specific choice of the vector $\boldsymbol{\beta}$, towards features that would cause the highest gain/lowest drop in stability if they were to be selected. Constant terms put aside[7], each feature influences (negatively) the total stability by its variance in the selection $s_f^2 \propto p_f(1-p_f)$. Feature f is given an attractiveness score sc_f, expressed in equation (6).

$$sc_f = \frac{(N+1)^2}{N} * (p_{f,no}(1-p_{f,no}) - p_{f,yes}(1-p_{f,yes})) \tag{6}$$

with $s_{f,no}^2$ the selection variance of feature f assuming f is not selected in the current task and $s_{f,yes}^2$ its selection variance if it were to be selected. N is the number of tasks for which feature sets have already been selected. This score is thus proportional to the difference of stability between the cases where the feature is selected for a given task and not. This is illustrated in table 1a where the current task is $T4$. For instance, the selection of the feature $F2$ in task $T4$ would make its mean selection, p_f, equal to 0.75. If it were not to be selected, p_f would be equal to 0.5. The attractiveness score of $F2$, sc_{F2} is actually positive, meaning that the selection of $F2$ in $T4$ would increase the measured stability.

The $\frac{(N+1)^2}{N}$ factor of equation (6) is there to correct a downards tendency of sc_f when the index of the considered task increases. This is illustrated on table 1b. If feature f is selected in each task, sc_f would actually decrease which would decrease the bias. It can be shown that including the correction term leads to $sc_f = 2 * p_f - 1$ with p_f the proportion of the selections of feature f in the past N tasks. Let S be the sum of the such selections. By definition, $p_f = \frac{S}{N}$,

[6] Tasks can be ranked naturally from their chronological order or by the domain expert.

[7] We purposely drop the $M/(M-1)$ term, for convenience. Also, the denominator $\frac{k}{d}(1-\frac{k}{d})$ is constant if the number k of selected features is fixed.

Table 1. Illustration of the attractiveness score (a). Need for the correction term of equation (6)(b).

(a)

	F1	F2	F3	F4	F5
T1	0	0	1	1	0
T2	0	1	1	0	1
T3	0	1	1	1	0
T4	?	?	?	?	?
$p_{f,no}$	0	0.5	0.75	0.5	0.25
$p_{f,yes}$	0.25	0.75	1	0.75	0.5
sc_f	-1	1/3	1	1/3	-1/3

(b)

	T1	T2	T3	T4	T5
f	1	1	1	1	
$s^2_{f,no}$		1/4	2/9	3/16	4/25
$s^2_{f,yes}$		0	0	0	0

$s^2_{f,no} = \frac{S}{N+1} * (1 - \frac{S}{N+1})$ and $s^2_{f,yes} = \frac{S+1}{N+1} * (1 - \frac{S+1}{N+1})$. Thus,

$$sc_f = \frac{(N+1)^2}{N} * \left(\frac{S}{N+1} - \frac{S^2}{(N+1)^2} - \frac{S+1}{N+1} + \frac{(S+1)^2}{(N+1)^2}\right) =$$

$$\frac{(N+1)^2}{N} * \left(\frac{2S+1}{(N+1)^2} - \frac{1}{N+1}\right) = \frac{(N+1)^2}{N} * \frac{2S-N}{(N+1)^2} = 2p_f - 1 \quad \square$$

This results demonstrates the intuitive idea that the selection should be biased towards features that have been selected often in previous tasks. Based on the attractiveness scores, we propose to pose

$$\beta_f = exp(-sc_f * \alpha_t) \tag{7}$$

to bias the selection towards previously selected features.[8] With $\alpha_t = 0$, features are learned independently on each task. On the contrary, an increasing α_t raises the bias towards features that were already selected in past tasks. Domain experts can thus tune the α_t values to control the accuracy-stability trade-off in such a transfer learning setting.

5 Experiments

In this section, we evaluate to what extent an actual compromise between prediction accuracy and selection stability can be made with the proposed approaches. Experiments are performed on two distinct tasks, prostate cancer diagnosis from microarray data and handwritten digit recognition. The Prostate dataset contains 12600-dimensional (microarray) gene expression data from 52 patients with prostate tumors and 50 healthy patients [22]. The Gisette dataset contains 5000-dimensional integer data, with features aimed at discerning pictures of the number 4 from the number 9. Gisette was originally constructed from the MNIST

[8] Again, $\boldsymbol{\beta}$ is post-centered such that $\mu(\boldsymbol{\beta}) = 1$ at each iteration of the RFE algorithm.

data but was extended with 2500 noisy features [9]. It consists of 6000 examples, but, in order to better illustrate the trade-off, only 100 examples are used here.

5.1 Evaluation Methodology

To obtain the results presented in the next sections, the following methodology has been used. Each (a, ϕ) pair is obtained with a different α (problem 1) or α_t (problem 2). For the single task stability problem, the β_f are first sampled from the gamma distribution. Then, M bootstrap samples are built. k features are then selected using the proposed biased RFE on each bootstrap sample. For the transfer learning stability problem, a single bootstrap sample for each task is created. Features are selected from it, then $\boldsymbol{\beta}$ for the next task is computed according to equation (7). The final prediction model is learned by minimizing the classical, unbiased, logistic loss with a L2 regularization (see equation (5)) with a non-strongly fitted[9] regularization parameter λ. Every model is evaluated on its out-of-bag examples. The mean accuracy as well as the stability of the selected features are computed. As these values are obviously dependent on the sampling of $\boldsymbol{\beta}$ (problem 1) or the features learned on the first few tasks (problem 2), this procedure is repeated B times and the mean values are reported. The 95% confidence *regions* of the expected value of the accuracy-stability trade-off are computed as well as the confidence interval on DA described in section 3. Stability of the feature selection (x-axis on Figures 2, 3 and 4) has not to be confused with its corresponding uncertainty which is the width of the confidence regions along the x-axis.

5.2 Single Task Selection Stability

The λ meta-parameter of the RFE formulation (equation (5)) has not been strongly optimized. A value of 0.1 which provides a good accuracy has been used for both tasks. To obtain the below graphs, the methodology detailed in section 5.1 has been used with $M = 30, k = 20$ and $B = 100$.

Results on both data sets can be seen in Figure 3. The blue curves are obtained without any prior knowledge. The top-left point of each subgraph corresponds to the (accuracy,stability) trade-off obtained with the classical logistic RFE method. Following Pareto lines from left to right, the shape α of the gamma distribution decreases. This makes the biased logistic RFE departs from its unbiased version which raises stability but reduces classification performance. As the method fails to reach maximum stability, it was extended with the trivial point $(a_{rand}, 1)$, obtained by always returning the same arbitrary feature subset.[10] It

[9] Values used for λ are 0.1 for problem 1 and 1 for problem 2. The final classification algorithm does not influence the selection stability. It can thus be optimized to maximize the predictive accuracy only.

[10] It is actually impossible to reach a maximum stability of 1 for a finite regularization parameter λ. In such a case, even with no regularization, a feature is not guaranteed to be always selected.

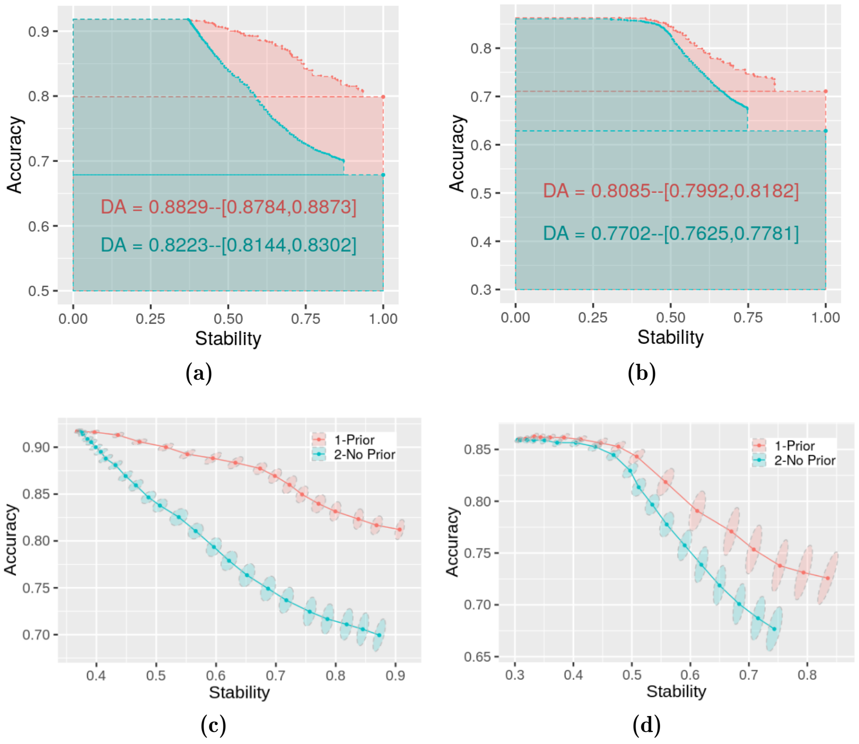


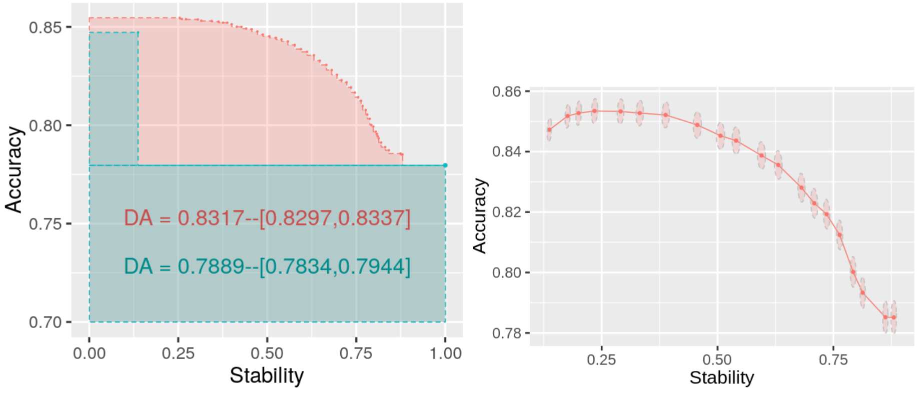
Fig. 3. Performance evaluation of the single task stability problem. DA obtained on Prostate (a,c) and Gisette (b,d) with or without prior knowledge and the corresponding confidence regions.

seems that, while it is possible to increase significantly the stability without degrading too much the accuracy on the Gisette dataset (Figure 3b), it is not the case for the Prostate dataset (Figure 3a) where the accuracy drops directly.

To measure the effect of prior knowledge, $N = 10$ examples are sampled randomly. The 100 features with the highest variance are selected as part of the prior knowledge, here representing a set of potentially relevant features. As can be seen on Figure 3a and 3b, even such a small prior knowledge improves the Dominance Area considerably.

Figures 3c and 3d have been obtained with a small subset of the Pareto points. The ellipses are the 95% confidence *regions* of the expected value (on the $\boldsymbol{\beta}$ sampling) of the accuracy-stability trade-off. For large α values, the importance of $\boldsymbol{\beta}$ is reduced, and thus the uncertainty limited. As α decreases, this confidence region grows. The ellipses are also all inclined towards the right. This represents the covariance between the accuracy and stability for a single $\boldsymbol{\beta}$ sampling. If the sampling appears to be bad, *.i.e.* poor features are prioritized, poor accuracy and poor stability are obtained. The opposite is true for a good sampling. By using the top-right and bottom-left point of each ellipse, it is possible to derive a confidence interval on the true DA of the method on these datasets.

5.3 Multi-task Selection Stability via Transfer Learning



(a) **(b)**

Fig. 4. Accuracy-stability trade-off in a transfer learning setting evaluated on Prostate (a). In red is displayed the DA obtained with the proposed biased RFE. In blue is the DA obtained by combining the two trivial options: either select features on each task independently, or always return the features selected for the first task. Confidence regions of a few points computed by the biased RFE in the transfer learning setting (b).

To generate different, yet similar, classification tasks, normally distributed noise has been added to the Prostate dataset. This noise is centered on 0 and has a specific standard deviation for every couple of feature and task, such that features relevant in some task, could be irrelevant in others. Yet, tasks are expected to share common informative features. 8 tasks are considered here, with an arbitrary order between them. Results with $k = 10, B = 500, \lambda = 1$ are shown on Figure 4a. The blue area is obtained by combining two trivial options. First, the features learned on the first task can be selected for all subsequent tasks, achieving a stability of 1. Or features can be learned independently from each other (equivalent to $\alpha_t = 0$). This strategy offers poor selection stability, but also a sub-optimal classification performance. Knowledge from previous tasks can be used to guide the search towards potentially good features for subsequent tasks. This increases both the accuracy and stability at first. Then, the accuracy starts to decrease, as the selection of features is forced too much. This tendency is better illustrated in Figure 4b, which contains some non-Pareto optimal points. This result is consistent with the conclusion drawn by the analysis of the Group-Lasso with ℓ_1/ℓ_p regularization [25], *i.e.* that weak coupling norms ($1.5 \leq p \leq 2$) outperforms no and strong coupling norms. Unlike for single task feature selection, the confidence regions are similar for all compromises, meaning that differential shrinkage does not increase the uncertainty of the obtained accuracy-stability pair. Furthermore, as the ellipses are straight, the measured accuracy and stability are uncorrelated.

6 Conclusion and Perspectives

The typical instability of standard feature selection methods is a key concern nowadays as it reduces the interpretability of the predictive models as well as the trust of domain experts towards the selected feature subsets. Such experts would often prefer a more stable feature selection algorithm over an unstable and slightly more accurate one. In this paper, the compromise between feature relevance and selection stability is made explicit by biasing the selection towards some features through differential shrinkage of the Recursive Feature Elimination algorithm. Domain experts are given the opportunity to select any Pareto-optimal trade-off of accuracy and selection stability based on their preferences. We propose the use of the hypervolume metric to assess the performance of methods realizing such a compromise. An associated confidence interval, based on the derivation of confidence *regions* of the accuracy-stability trade-off, is derived.

Results on prostate cancer diagnosis and handwritten recognition tasks show that the selection stability can be increased at will, often with a cost of classification performance. When some prior knowledge is available, far better compromises can be made. The design and evaluation of hybrid methods, learning the prior knowledge from the data, and using it to stabilize the selection is part of our future work.

Motivated by the needs of domain experts, across tasks feature stability is also studied in a transfer learning setting (*i.e.* when tasks are ordered). A biasing scheme that takes the stability measure explicitly into account is proposed. For similar, yet different, tasks, we show on microarray data that some bias is at first beneficial for both the accuracy and the stability. A too strong bias continues to increase the selection stability but at the cost of some classification performance, as the most relevant features vary across tasks. Our approach is evaluated here in a simulated transfer learning setting and further experimental validations will be conducted.

Different multi-task feature selection methods have been proposed in the literature (*e.g.* Group-Lasso with ℓ_1/ℓ_p regularization [25], additive linear models [8], ...). Such methods were introduced with the primary objective of building accurate predictive models across several (this time unordered) tasks. We will study to which extent they could also be used to allow the tuning of the across task selection stability and classification performance trade-off. The biased RFE proposed here can be extended to tackle classical multi-task feature selection, for example by prioritizing the most relevant features when all tasks are considered together. Our future work includes the evaluation of all these approaches in the proposed assessment framework.

The present paper answers the growing necessity of considering the selection stability not only as a side-effect of learning accurate predictive models but as an actual goal in a bi-objective framework. It proposes initial approaches to learn Pareto-optimal compromises in such a framework and, hopefully, opens the way to new works and improvements in this area.

References

1. Abeel, T., Helleputte, T., Van de Peer, Y., Dupont, P., Saeys, Y.: Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. Bioinformatics **26**(3), 392–398 (2010). https://doi.org/10.1093/bioinformatics/btp630
2. Alelyani, S.: On feature selection stability: A data perspective. Ph.D. thesis, Arizona State University (2013)
3. Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. In: Advances in neural information processing systems. pp. 41–48 (2007)
4. Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. Machine Learning **73**(3), 243–272 (2008)
5. Awada, W., Khoshgoftaar, T.M., Dittman, D., Wald, R., Napolitano, A.: A review of the stability of feature selection techniques for bioinformatics data. In: Information Reuse and Integration (IRI), 2012 IEEE 13th International Conference on. pp. 356–363. IEEE (2012)
6. Boulesteix, A.L., Slawski, M.: Stability and aggregation of ranked gene lists. Briefings in bioinformatics **10**(5), 556–568 (2009)
7. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. Journal of bioinformatics and computational biology **3**(02), 185–205 (2005)
8. Evgeniou, T., Pontil, M.: Regularized multi–task learning. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 109–117. ACM (2004)
9. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A.: Feature extraction: foundations and applications, vol. 207. Springer (2008)
10. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine learning **46**(1-3), 389–422 (2002)
11. Han, Y., Yu, L.: A variance reduction framework for stable feature selection. Statistical Analysis and Data Mining: The ASA Data Science Journal **5**(5), 428–445 (2012)
12. Helleputte, T., Dupont, P.: Feature selection by transfer learning with linear regularized models. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 533–547. Springer (2009)
13. Helleputte, T., Dupont, P.: Partially supervised feature selection with regularized linear models. In: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 409–416. ACM (2009)
14. Kalousis, A., Prados, J., Hilario, M.: Stability of feature selection algorithms. In: Data Mining, Fifth IEEE International Conference on. pp. 8–pp. IEEE (2005)
15. Kuncheva, L.I.: A stability index for feature selection. In: Artificial intelligence and applications. pp. 421–427 (2007)

16. Liu, H., Palatucci, M., Zhang, J.: Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 649–656. ACM (2009)
17. Nogueira, S., Sechidis, K., Brown, G.: On the stability of feature selection algorithms. The Journal of Machine Learning Research **18**(1), 6345–6398 (2017)
18. Obozinski, G., Taskar, B., Jordan, M.: Multi-task feature selection. Statistics Department, UC Berkeley, Tech. Rep **2** (2006)
19. Obozinski, G., Taskar, B., Jordan, M.I.: Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing **20**(2), 231–252 (2010)
20. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics **23**(19), 2507–2517 (2007)
21. Shi, L., Reid, L.H., Jones, W.D., Shippy, R., Warrington, J.A., Baker, S.C., Collins, P.J., De Longueville, F., Kawasaki, E.S., Lee, K.Y., et al.: The microarray quality control (maqc) project shows inter-and intraplatform reproducibility of gene expression measurements. Nature biotechnology **24**(9), 1151 (2006)
22. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., et al.: Gene expression correlates of clinical prostate cancer behavior. Cancer cell **1**(2), 203–209 (2002)
23. Somol, P., Novovicova, J.: Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(11), 1921–1939 (2010)
24. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) pp. 267–288 (1996)
25. Vogt, J., Roth, V.: A complete analysis of the l_1, p group-lasso. arXiv preprint arXiv:1206.4632 (2012)
26. Zhang, M., Zhang, L., Zou, J., Yao, C., Xiao, H., Liu, Q., Wang, J., Wang, D., Wang, C., Guo, Z.: Evaluating reproducibility of differential expression discoveries in microarray studies by considering correlated molecular changes. Bioinformatics **25**(13), 1662–1668 (2009)
27. Zhang, Y., Yang, Q.: A survey on multi-task learning. arXiv preprint arXiv:1707.08114 (2017)
28. Zitzler, E., Brockhoff, D., Thiele, L.: The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In: International Conference on Evolutionary Multi-Criterion Optimization. pp. 862–876. Springer (2007)
29. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE transactions on Evolutionary Computation **3**(4), 257–271 (1999)
30. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **67**(2), 301–320 (2005)

Deep Bayesian Semi-Supervised Active Learning for Sequence Labelling

Tomáš Šabata[1], Juraj Eduard Páll[2], and Martin Holeňa[3]

[1] Faculty of Information Technology, Czech Technical University in Prague,
Prague, Czech Republic
tomas.sabata@fit.cvut.cz

[2] Faculty of Mathematics and Physics, Charles University,
Prague, Czech Republic
palljuraj1@gmail.com

[3] Institute of Computer Science of the Czech Academy of Sciences,
Prague, Czech republic
martin@cs.cas.cz

Abstract. In recent years, deep learning has shown supreme results in many sequence labelling tasks, especially in natural language processing. However, it typically requires a large training data set compared with statistical approaches. In areas where collecting of unlabelled data is cheap but labelling expensive, active learning can bring considerable improvement. Sequence learning algorithms require a series of token-level labels for a whole sequence to be available during the training process. Annotators of sequences typically label easily predictable parts of the sequence although such parts could be labelled automatically instead. In this paper, we introduce a combination of active and semi-supervised learning for sequence labelling. Our approach utilizes an approximation of Bayesian inference for neural nets using Monte Carlo dropout. The approximation yields a measure of uncertainty that is needed in many active learning query strategies. We propose Monte Carlo token entropy and Monte Carlo N-best sequence entropy strategies. Furthermore, we use semi-supervised pseudo-labelling to reduce labelling effort.
The approach was experimentally evaluated on multiple sequence labelling tasks. The proposed query strategies outperform other existing techniques for deep neural nets. Moreover, the semi-supervised learning reduced the labelling effort by almost 80% without any incorrectly labelled samples being inserted into the training data set.

Keywords: Active Learning · Semi-supervised Learning · Bayesian Inference · Deep Learning · Sequence Labelling

1 Introduction

Deep learning is achieving state-of-the-art performance in image or video processing, audio processing or natural language processing. However, without using a pretrained model, deep learning typically requires a large amount of data. To obtain unlabelled input for deep networks in video processing, cameras and other sensors are increasingly available. In natural language processing, a lot of unlabelled inputs can be obtained for almost no cost by gathering them from web sites. Unfortunately, labelling such data is very time consuming and expensive.

In this situation, we can benefit from semi-supervised learning using a large unlabelled dataset along with a small labelled one. Another option is to use active learning wherein each iteration, a part of an annotation budget is spent on labelling the most informative unlabelled samples. The model is retrained including those new samples and the process repeats. The annotation budget is significantly lower than the total number of available unlabelled samples.

Although active learning is a promising way to benefit from unlabelled data, the most common query strategy, uncertainty sampling, requires a measure of uncertainty. In sequence labelling, the measure can be easily defined for statistical models, such as hidden Markov models or conditional random fields (CRF), as they provide a probability of the labelled sequence or a marginal probability distribution for each element of the sequence. For neural networks, defining an uncertainty measure is more complicated since the soft-max activation function, typically used in the last network layer, does not correspond to a real uncertainty of network predictions. To overcome this issue, one can use a Bayesian neural network or include a statistical model, such as CRF, as the last layer of the network.

In sequence labelling, query strategies can be divided into two groups. The first group computes the uncertainty of the sequence predicted by a model. Query strategies of the second group compute uncertainties of separated tokens and then aggregate them to express the uncertainty of the whole sequence.

Querying the most informative sequence means that the annotator has to label every token of the sequence. This is expensive and often not necessary because some tokens can be very reliably annotated automatically. This situation can be found in many natural language processing (NLP) tasks, where some words can be assigned to only one category and we can predict that without knowing the context. A similar situation can be found in a video where two consecutive frames often contain the same or similar information and labelling all frames might be inefficient.

In this paper, we propose an active learning algorithm for sequence labelling with deep neural networks that queries labels of the most informative tokens whereas other labels are labelled automatically.

In the following section, we summarize approaches addressing this topic. In section 3, we define the architecture of our sequence labelling models. In section 4, we describe details of the proposed algorithm. The algorithm is evaluated with experiments on tasks from natural language processing and the results are shown in section 5.

2 Related Work

Sequence labelling models have been used in many areas such as part of speech tagging (POS) or named entity recognition (NER) [25], handwritten recognition [9], protein secondary structure prediction [19], video analysis [39] or facial expression dynamic modeling [4]. In the early years, probabilistic models were the most frequent approach. The most commonly used among them are Hidden Markov models, dynamic Naive Bayesian classifiers, maximum entropy Markov models or Conditional Random fields.

With the increasing amount of data and computational power, and with formulating new network topologies, deep networks are more and more popular in sequence labelling. This is especially true for long short term memory networks (LSTM), which deal well with vanishing gradient problem and are able to incorporate context far from the predicted token. One of the state-of-the-art topologies in sequence labelling is the bi-directional LSTM network (BI-LSTM) [34] or an extended version with a CRF layer on top (BI-LSTM-CRF) [15]. Another interesting topology specific to language processing uses an additional layer (LSTM [23] or CNN [22]) as a character-level embedding for words.

Active Learning in Sequence Labelling was studied intensively for probabilistic models [37]. Query strategies used in AL can be categorized into several groups. Uncertainty Sampling that selects the most uncertain samples, Query by Committee selects samples in which a committee disagree the most, Expected Gradient Length selects samples that would conduct the greatest change to the current model or Fisher information strategy that selects samples that minimize the model variance. These strategies differ in computational complexity and model requirements. The most commonly used strategy, uncertainty sampling, requires the model to return confidence of its predictions. Furthermore, to avoid querying samples that are rather outliers than representative samples, the informativeness of the sample is weighted by its average similarity to all other samples. The technique is called information density [37].

In active learning for sequence labelling, the most informative sequence is labelled. The sequence is then added to the training set, the model is retrained and the process repeats. This requires the whole sequence to be labelled at once. In contrast, Tomanek [40] introduced the *SeSAL* algorithm, where parts of sequences can be labelled automatically. That algorithm was designed for HMMs and CRFs.

Active Learning in Connection with Deep Learning Although active learning has been applied to many ML tasks, application to deep learning is marginal compared to probabilistic modelling. One of the main problems in deep active learning is that many query strategies require some uncertainty estimate, however, most kinds of deep neural networks rarely support it. In literature, we can find several approaches approximating the model posterior: variational inference [8], probabilistic back-propagation [11], Monte-Carlo (MC) dropout [6, 16] or mixture density networks [3]. With such approximations of uncertainty, active learning has been used in connection with deep learning in image classification [7] or text classification [1]. In the sequence labelling area, active deep learning was successfully used for NER. In [38], a CNN-CNN-LSTM network was used together with active learning in a setup where the whole queried sequence had to be labelled at once.

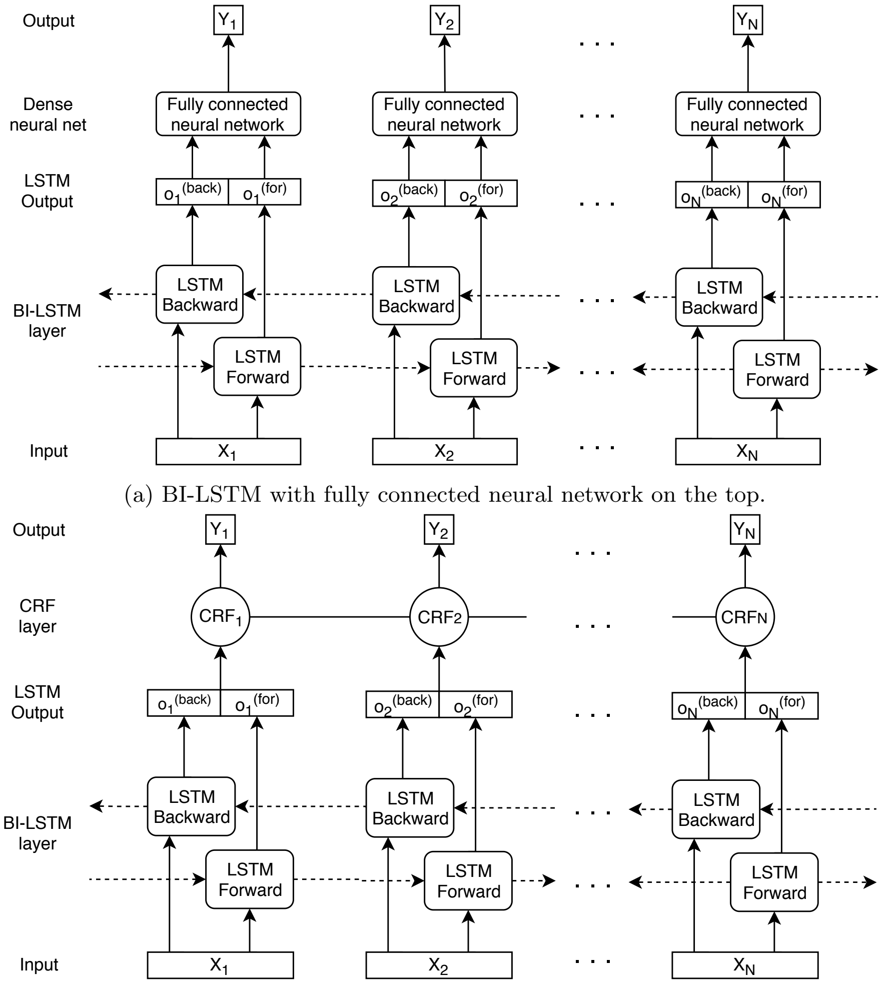
3 Underlying Models

A sequence labelling model assigns categorical labels to all members (tokens) of a sequence of observed values. In general, it considers the optimal label for a given token to be dependent on the choices of nearby tokens. The problem is often simplified through the assumption that the sequence of labels is a Markov chain. With that simplification, the problem can be modelled with a probabilistic graphical model such as a hidden Markov model [29] or a conditional random field. Although the probabilistic models work well on many sequence labelling tasks [21], the Markov property assumption might be too restrictive and unrealistic for problems where a wider context is needed to label tokens correctly. This can be overcome by considering dependencies of higher-order but the computational complexity is growing exponentially with the order which makes these models unusable for real-world problems. Deep learning neural networks can help to overcome the issue of wider context.

Deep Learning Models. In sequence labelling, various kinds of neural networks are used. These networks are typically designed for a specific task. This is particularly true for their first layers that extract features. In NLP, a character level embedding layer extracts low-level features from the text. In video analysis, feature vectors are extracted using pretrained convolutional networks. After the first layer, a layer that incorporates contextual information from neighbouring elements is plugged in. The most commonly used layers on this level are LSTM cells [13] or gated recurrent unit (GRU) [2]. Last, a sequence decoder layer is used to predict the final sequence. Both context independent layers, for example a fully connected dense layer (BI-LSTM-FCN) (Figure 1a), and contextual layers, for example conditional random fields (BI-LSTM-CRF) (Figure 1b), can be used.

Moreover, to avoid over-fitting, a dropout regularization technique can be used. In our experiments, we use dropout for non-recurrent connections (solid lines in Figure 1). The dropout enabled for each layer allows to estimate prediction uncertainties, as the following section describes.

Bayesian neural networks aim to tackle several drawbacks of neural networks such as overconfidence about their predictions or tendency to overfitting. In classification, the prediction probabilities obtained from the soft-max function are often erroneously interpreted as model confidence [6]. It means, the model



(a) BI-LSTM with fully connected neural network on the top.

(b) BI-LSTM with conditional random fields on the top.

Fig. 1: Schematic representation of deep neural network sequence labelling models used in experiments.

can be uncertain despite high values of the soft-max function and these values require correct calibration [10] before using them as confidences. The main idea of Bayesian neural networks is placing probabilistic distribution over nets' weights [24,26]. However, the approach introduces two issues, intractable inferences and computation costs. Although stochastic variational inference [14,18,27,30] solves the problem with intractable inference, the number of parameters is doubled and it requires more time to converge.

Gal & Ghahramani introduced *Monte Carlo dropout* [6]. They have shown that dropout or various other stochastic regularization techniques can be used to obtain an approximation of Bayesian inference. Consider a sequence of input vectors denoted x to which a sequence of labels denoted y is assigned. A training set containing pairs $\langle x, y\rangle$ is denoted T. Consider a neural net with parameters ω that uses dropout at every layer for the training. Using dropout during testing can be seen as sampling from a model's approximate posterior. This leads to approximate variational inference in which a tractable distribution $q^*_\theta(\omega)$ minimizes the Kullback-Leibler (KL) divergence [20] to the true model posterior $p(\omega|T)$ given a training set T. The prediction uncertainty can be approximated by marginalization over the approximate posterior using Monte Carlo integration:

$$\begin{aligned} p(y = c|x, T) &= \int p(y = c|x, \omega)p(\omega|T)d\omega \\ &\approx \frac{1}{R}\sum_{t=1}^{R} p(y = c|x, \hat{\omega}_t), \end{aligned}$$

where $\hat{\omega}_t \sim q^*_\theta(w)$, R is the number of Monte Carlo runs, and where $q_\theta(w)$ denotes the Dropout distribution [7].

Monte Carlo dropout does not affect the model training complexity, however, each point has to be inferred repeatedly to obtain prediction uncertainty.

4 Active Learning Strategies

Query strategies for sequence labelling models can be divided into several frameworks such as uncertainty sampling (US), query by committee (QbC), expected gradient length (EGL) or information density (ID) [36]. In this section, we describe some of the strategies and propose how they can be used together with the introduced models. The most informative samples are considered to be found by maximising a particular utility function:

$$x^* = \arg\max_x \phi(x).$$

The probability of the sequence y in a by model M given the input sequence x is denoted $P_M(y|x)$. The set of labelled sequences is denoted $\mathcal{L}$ and set of unlabelled sequences is denoted $\mathcal{U}$.

4.1 Query Strategies Utility Functions

Query strategies of the uncertainty sampling framework select the sequences that have the most uncertain label. The uncertainty measure can be expressed in several ways. **Least confidence** query strategy [5] selects the sequence with the lowest probability of the most likely sequence:

$$\phi^{LC}(x) = 1 - P_M(y^*|x),$$

where y^* is the most likely sequence. For CRF, the most likely sequence and its probability can be found using the Viterbi algorithm. For neural nets, the probability of the most likely sequence can be approximated by an empirical probability based on Monte Carlo dropout, which will be denoted $P_M^{\mathrm{MC}}(y|x)$. This empirical distribution is calculated by counting the occurrences of the sequence y for input sequence x in several forward passes through the network, where each forward pass has a different dropout mask. These counts are normalized to sum to 1.

Margin query strategy [33] selects samples where the first and the second most likely sequences have the most similar probabilities. Finding the second most likely sequence in case of probabilistic graphical models requires an updated version of the Viterbi algorithm called N-best Viterbi algorithm [35]. For neural nets, the distribution $P_M^{\mathrm{MC}}(y|x)$ can be used to find the probability of the second most likely path.

The margin query strategy utility function is defined as:

$$\phi^M(x) = -\big(P_M(y_1^*|x) - P_M(y_2^*|x)\big),$$

where y_1^* and y_2^* are first and second the most likely sequences.

Token entropy query strategy [37] uses the Shannon entropy of the model's posteriors:

$$H_M(l) = -\sum_k^K P_M(y_l = k) \log P_M(y_l = k),$$

where y_l is label of the sequence in time l and K is the number of possible labels, over its labellings to define the utility function for selecting the most uncertain sequence:

$$\phi^{TE}(x) = -\frac{1}{L}\sum_{l=1}^{L} H_M(l),$$

where L is the length of the sequence. The utility function is normalized by the length of the sequence. Omitting this normalization, the strategy would lead to querying long sequences as they contain more information. The unnormalized utility function is called **total token entropy**.

Whereas the marginal probability for CRF can be calculated using forward and backward scores, those scores are not available for neural networks. We propose an approximation called **Monte Carlo approximation token entropy**, which uses the idea of Bayesian inference with Monte Carlo [6]:

$$\phi_{\mathrm{MC}}^{TE}(x) = -\frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{K}\big(P_M^{\mathrm{MC}}(y_l = k) \log P_M^{\mathrm{MC}}(y_l = k)\big).$$

Sequence entropy query strategy computes the entropy of probabilities of all possible sequences. This strategy is unfeasible for long sequences as the number of possible sequences grows exponentially with the length of the sequence. Furthermore, it is not possible to obtain probabilities of a particular sequence directly in neural network based models. For probabilistic graphical models, the strategy can be approximated with the N-best sequence entropy [17]:

$$\phi^{NSE}(x) = -\frac{1}{C_1}\sum_{\hat{y}\in\mathcal{N}} P_M(\hat{y}|x)\text{log}P_M(\hat{y}|x),$$

where $\mathcal{N} = \{y_1^*, ..., y_N^*\}$ is set of N most likely sequences found by N-best Viterbi algorithm [35] and C_1 normalizes the probabilities to sum to 1.

While the probabilities of N most likely sequences can be obtained directly in probabilistic models, this cannot be done in neural networks. Therefore, we propose a **Monte Carlo approximation of the sequence entropy**:

$$\phi_{\text{MC}}^{NSE}(x) = -\frac{1}{C_2}\sum_{\hat{y}\in\mathcal{N}^{\text{MC}}} P_M^{MC}(\hat{y}|x)\text{log}P_M^{MC}(\hat{y}|x), \tag{1}$$

where $\mathcal{N}^{\text{MC}} = \{y_1, y_2, \dots\}$ is set of all sequences predicted by Monte Carlo sampling and C_2 normalizes the probabilities to sum to 1.

In the query by committee framework, a committee of models $\mathcal{C} = \{M^{(1)}, ..., M^{(C)}\}$, representing different hypotheses, is maintained during the whole process of learning. The committee is used to query the sequence over which the members are most in disagreement about how to label it. The committee is usually trained using bagging. In each round, the labelling set is sampled with replacement to create a unique training set $L^{(C)}$ that is used to train model $M^{(C)}$. The committee prediction is obtained by models voting. In the context of deep neural networks, maintaining a committee is too expensive for practical use. Although dropout can be considered as a form of bagging [12], we do not deal with the framework in the paper.

US and QbC strategies are prone to querying outliers as they are often uncertain for the model and the committee of models often disagrees about them. The framework, called **information density**(ID), can be used to avoid this problem. ID uses a base utility function $\phi_B(x)$ and weights it by samples‘ representativeness. All above defined utility functions can be used as base utility functions. ID utility function is defined:

$$\phi^{ID}(x) = \phi_B(x) \times \Big(\frac{1}{|\mathcal{U}|}\sum_{u=1}^{|\mathcal{U}|}\text{sim}(x, x^{(u)})\Big)^{\beta}, \tag{2}$$

where $\text{sim}(x, x^{(u)})$ is a chosen similarity function for two sequences and β a parameter that controls a relative importance of the representativeness term. The similarity measure differs from task to task.

4.2 Token-Level Semi-Supervised Active Learning

In the standard AL approach, the annotator has to label the whole sequence although the sequence can contain subsequences that do not add too much value to the utility function. If the model is sufficiently learned, these subsequences can be easily annotated automatically using model inference. The decision whether a token can be labelled automatically can rely on some kind of model confidence [40] or the disagreement about the most probable paths [31].

We propose to use a combination of active and semi-supervised learning. For models with CRF layer on the top, marginal probability represents the model prediction confidence. Otherwise, the Monte Carlo dropout estimates the model prediction confidence. First, the most informative sequence is found with a chosen query strategy. Tokens in which the model is confident are automatically labelled using semi-supervised learning and the rest is given to an annotator. The labelled sequence is added to the training set and the process repeats. Details of the approach are described in Algorithm 1. The confidence threshold has to be chosen according to the model, problem type and query strategy.

Algorithm 1: Sequential semi-supervised AL framework

```
Input:
$\mathcal{L}$: labelled set
$\mathcal{U}$: unlabelled set
$\phi(\cdot)$: query strategy utility function
$\theta$: confidence threshold
$M$: model type
begin
    train model $m$ of type $M$ on data set $L$
    while stopping criterion is not met do
        // Find the most informative sequence from $\mathcal{U}$
        $x^* = \text{argmax}_{x \in \mathcal{U}} \phi(x)$
        // label the sequence with the model or query the annotator
        $\hat{y} = m(x^*)$
        for $i = 1$ to length of $x^*$ do
            if $P_m(y_i = \hat{y}_i | x^*) > \theta$ then
                $y_i^* = \hat{y}_i$
            else
                $y_i^* = \text{query}(x_i^*)$
            end
        end
        $\mathcal{L} = \mathcal{L} \cup \langle x^*, y^* \rangle$
        $\mathcal{U} = \mathcal{U} \setminus x^*$
        retrain model $m$ on $\mathcal{L}$
    end
end
```

5 Experiments

To evaluate the performance of the proposed approach, we have chosen three different sequence labelling problems: named entity recognition(NER), part of speech tagging (POS) and chunking. Experiments were performed with two sequential models: BI-LSTM-FCN with Monte Carlo dropout and BI-LSTM-CRF, and various query strategies designed for each of those models.

In the paper, we report two experiments. The first experiment tests proposed query strategies against random sampling and least confident query strategies as a baseline. The second experiment is using sequential semi-supervised active learning framework to reduce the labelling effort. Our primary aim was reducing the amount of labelled data required for training, rather than labelling performance. Therefore, we did not extensively optimize hyper-parameters such as learning rate, batch size or momentum.

5.1 Experiment Design

The experiments were performed on the publicly available benchmark dataset CoNLL 2003 [32]. The dataset provides a predefined training set and two testing sets for POS, NER and Chunking. We report performance for the testing set A. The training set was randomly divided into a labelled set and an unlabelled set in the ratio 1:9.

Both models use GloVe embeddings [28] where each word is represented by a vector of length 300. The models contain two LSTM hidden layers with size 100 and dropout with probability 0.4 applied to all layers. The last layer of the BI-LSTM-FCN is linear and uses the soft-max activation function. The number of forward passes for computing P_{MC} was set to 500.

First, each model was trained on the labelled training set with 10% of the original size for 30 epochs. This model was used for experiments with all query strategies. For each query strategy, the model was used to find the most likely paths and their scores together with tokens prediction confidences for all unlabelled sequences. The most informative sequences were selected and annotated until the annotation budget was exhausted. We have defined the annotation budget of one AL cycle in two ways: the number of labelled sequences and the total number of annotated tokens. In the first scenario, 100 sequences were selected and annotated, whereas, in the second scenario, sequences were annotated until the total number of annotated tokens reached 1000. With the updated labelled training set, the model was updated by iterative training for one epoch, then the new score was calculated. This active learning cycle was repeated 20 times. In the second experiment, samples were sorted according to their confidences, and the threshold value was chosen to achieve 0% or 1% of incorrectly labelled samples.

Early results showed that proposed query strategies are prone to select outliers. Therefore, the information density wrapping strategy was used for all of them. Each sequence was represented by the average of embedding vectors. The representativeness of the sequence was computed as an average cosine distance to all other sequences in the unlabelled dataset. The cosine distance is claimed to be an efficient similarity measure of the linguistic or semantic similarity of corresponding words for the chosen embedding [28]. In the results, we use the names of base query strategies for clarity.

5.2 Results

In experiments, we studied the achieved performance in terms of F-measure (specifically F1 score) and accuracy by particular query strategies and the number of tokens that can be labelled automatically by semi-supervised learning. We report the macro-averaged F1 score that is calculated:

$$F1_{\text{macro}} = \frac{1}{|Q|} \sum_{q \in Q} F1_q,$$

where Q is the set of all possible labels and $F1_q$ is the F1 score for the class labeled q considered as the positive class and all remaining classes as the negative class.

These scores were compared with the models learned on the whole labelled dataset and models learned on a small labelled dataset that was later used for active learning.

Query Strategies Comparison Query strategies were compared in two AL scenarios: an unlimited number of tokens and a limited number of tokens. Table 1a shows that the strategies MC total token entropy and MC sequence entropy outperforms other strategies in both F1 score and accuracy. The MC total token entropy, however, required more tokens to be labelled. In NER, it queried almost twice as many tokens. In the scenario with a limited number of tokens, the MC sequence entropy dominates over other strategies except in the Chunking.

Table 1b shows that for the BI-LSTM-CRF model, the least confident and total token entropy query strategies have shown better results compared to the token entropy query strategy. We conclude that total token entropy query strategy dominates in the scenario with an unlimited number of tokens, whereas least confident achieves better results in the scenario with a limited number of tokens. The sequence entropy query strategy is missing as our implementation was lacking n-best Viterbi algorithm.

Moreover, Table 1b shows that the MC sequence entropy query strategy is the best among the compared strategies in the NER and POS tasks during the whole AL loop if the number of annotated tokens is limited in each cycle of the AL loop (Figures 2a and 2b).

Active Learning in Combination with Semi-supervised Learning Last, we studied a possible reduction of the labelling effort using semi-supervised learning. We report how many tokens were automatically labelled if the threshold is set to not allow errors propagate into the training dataset and if 1% of errors are allowed. The results in Table 2 indicate that the BI-LSTM-CRF model has a more reliable uncertainty measure for the marginal distribution than the BI-LSTM-FCN model. It can reduce the labelling effort up to almost 80% without any incorrectly labelled samples being inserted into the training data set. The labelling effort is reduced up to almost 84% with 1% of incorrectly labelled samples being inserted into the training dataset.

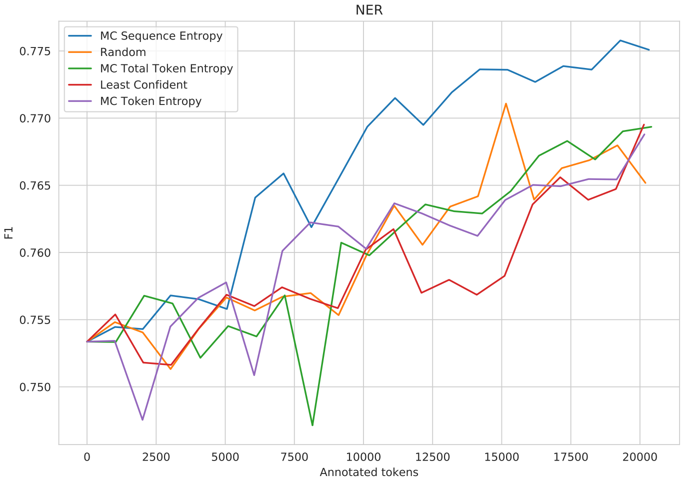
Table 1: Comparison of query strategies for BI-LSTM-FCN. The column 'Tokens' represents the ratio of labelled tokens to the number of all tokens in sequences from the complete dataset. The percentage sign is omitted. The first two lines of the table report performance of the supervised model trained on complete dataset and dataset with only 10% of training data available respectively.

(a) BI-LSTM-FCN

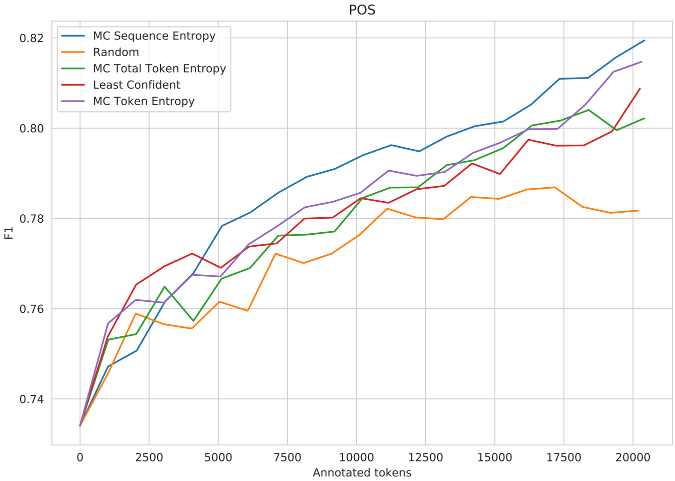
	NER			POS			Chunking		
	F1	Acc	Tokens	F1	Acc	Tokens	F1	Acc	Tokens
No active learning									
BI-LSTM-FCN	85.3	98.6	100	85.3	95.4	100	70.5	96.0	100
	75.4	97.7	10	74.6	92.0	10	53.7	93.7	10
Active learning with an unlimited number of tokens									
Random	73.5	98.1	24	79.6	93.2	24	54.8	94.7	25
Least Confident	76.2	98.0	17	83.2	93.7	39	57.8	95.0	33
MC Token Entropy	77.2	98.2	24	82.7	93.7	40	57.8	94.8	36
MC Total Token Entropy	**82.4**	98.3	40	83.0	93.7	45	**63.4**	95.0	45
MC Sequence Entropy	78.0	**98.4**	26	**83.3**	**94.0**	41	59.7	95.0	37
Active learning with a limited number of tokens									
Random	76.5	98.0	20	78.1	93.3	20	53.3	94.5	20
Least Confident	76.9	98.0	20	80.8	93.1	20	55.2	94.5	20
MC Token Entropy	76.8	98.1	20	81.5	93.0	20	55.3	94.5	20
MC Total Token Entropy	76.9	98.1	20	80.2	93.2	20	**56.7**	94.4	20
MC Sequence Entropy	**77.5**	**98.3**	20	**82.0**	**93.4**	20	55.3	94.4	20

(b) BI-LSTM-CRF

	NER			POS			Chunking		
	F1	Acc	Tokens	F1	Acc	Tokens	F1	Acc	Tokens
No active learning									
BI-LSTM-CRF	85.5	98.7	100	82.3	95.3	100	58.1	95.9	100
	75.9	97.8	10	72.8	92.0	10	50.2	93.7	10
Active learning with unlimited number of tokens									
Random	76.8	98.1	24	75.2	93.3	24	50.7	94.6	24
Least Confident	**78.4**	**98.6**	33	81.4	**94.0**	40	56.0	**95.2**	37
Token Entropy	77.3	98.3	23	75.0	93.0	17	55.9	94.9	20
Total Token Entropy	78.0	98.5	32	**82.0**	**94.0**	39	**56.3**	95.1	40
Active learning with limited number of tokens									
Random	76.4	98.0	20	75.6	93.2	20	55.7	94.4	20
Least Confident	77.4	**98.3**	20	**82.1**	93.3	20	**57.0**	94.5	20
Token Entropy	77.4	**98.3**	20	75.6	**93.6**	20	56.4	**94.7**	20
Total Token Entropy	**77.5**	**98.3**	20	79.3	93.3	20	56.5	94.6	20



(a) NER



(b) POS

Fig. 2: Query strategies comparison for NER and POS for BI-LSTM-FCN in the scenario with fixed number of annotated tokens.

6 Conclusions and Future Work

In this paper, we presented an application of Monte Carlo dropout, an approximation of Bayesian inference for deep neural networks, to active learning strate-

Table 2: Proportion of data labelled automatically by pseudo-labelling.

Task type		NER		POS		CHUNK	
Allowed errors		0%	1%	0%	1 %	0 %	1 %
BI-LSTM-FCN	Least confident	6.9	21.5	0.2	1.7	3.5	10.5
	Sequence entropy	14.2	28.7	0.3	2.0	3.6	10.5
	Total token entropy	7.8	12.8	0.2	1.4	2.0	6.2
	Token entropy	9.0	18.7	0.2	1.6	2.6	8.3
BI-LSTM-CRF	Least confident	77.3	84.3	72.4	82.5	66.6	79.6
	Token entropy	79.5	83.7	72.2	77.9	72.9	79.3
	Total token entropy	75.5	83.5	71.2	81.6	65.3	78.3

gies developed for probabilistic graphical models. We proposed two not yet used adaptations of token entropy and sequence entropy query strategies suitable for LSTM-type deep neural networks. Moreover, we tested a combination of active and semi-supervised learning for sequence labelling for that network.

The proposed query strategies have shown a substantial improvement over the until now used strategy in sequence labelling with deep neural networks, least confident. The proposed strategies outperformed the least confident in all three considered sequence labelling tasks in case of the network without a CRF layer. This is particularly true, if the annotation budget is limited for each active learning batch, which is a typical real-world situation.

The combination of active and semi-supervised learning allows us to achieve up to 80% labelling cost reduction for the BI-LSTM-CRF model. The uncertainty measure based on Monte Carlo dropout, however, still needs improvement to achieve labelling effort reduction comparable with BI-LSTM-CRF. To this end, we would like to study uncertainty measures provided by other approaches to Bayesian recurrent neural networks.

Although uncertainty sampling has shown to be applicable to deep neural networks, other active learning frameworks have not been enough studied. In the future, we would like to study, in the context of sequence labelling and deep neural networks, active learning based on expected gradient length. In addition to this, we would like to apply deep active learning to sequence labelling in video processing, where context is also very important information.

Acknowledgements

The work has been supported by the grant 18-18080S of the Czech Science Foundation (GACR).

References

1. Burkhardt, S., Siekiera, J., Kramer, S.: Semi-supervised bayesian active learning for text classification. In: Bayesian Deep Learning (2018)

2. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
3. Choi, S., Lee, K., Lim, S., Oh, S.: Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 6915–6922. IEEE (2018)
4. Cohen, I., Sebe, N., Garg, A., Chen, L.S., Huang, T.S.: Facial expression recognition from video sequences: temporal and static modeling. Computer Vision and image understanding **91**(1-2), 160–187 (2003)
5. Culotta, A., McCallum, A.: Reducing labeling effort for structured prediction tasks. In: AAAI. vol. 5, pp. 746–751 (2005)
6. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: international conference on machine learning. pp. 1050–1059 (2016)
7. Gal, Y., Islam, R., Ghahramani, Z.: Deep bayesian active learning with image data. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 1183–1192. JMLR. org (2017)
8. Graves, A.: Practical variational inference for neural networks. In: Advances in neural information processing systems. pp. 2348–2356 (2011)
9. Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: Advances in neural information processing systems. pp. 545–552 (2009)
10. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 1321–1330. JMLR. org (2017)
11. Hernández-Lobato, J.M., Adams, R.: Probabilistic backpropagation for scalable learning of bayesian neural networks. In: International Conference on Machine Learning. pp. 1861–1869 (2015)
12. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
14. Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J.: Stochastic variational inference. The Journal of Machine Learning Research **14**(1), 1303–1347 (2013)
15. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
16. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: Advances in neural information processing systems. pp. 5574–5584 (2017)
17. Kim, S., Song, Y., Kim, K., Cha, J.W., Lee, G.G.: Mmr-based active machine learning for bio named entity recognition. In: Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers. pp. 69–72. Association for Computational Linguistics (2006)
18. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
19. Krogh, A., Larsson, B., Von Heijne, G., Sonnhammer, E.L.: Predicting transmembrane protein topology with a hidden markov model: application to complete genomes. Journal of molecular biology **305**(3), 567–580 (2001)

20. Kullback, S.: Information theory and statistics. Courier Corporation (1997)
21. Lafferty, J., McCallum, A., Pereira, F.C.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data (2001)
22. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360 (2016)
23. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional lstm-cnns-crf. arXiv preprint arXiv:1603.01354 (2016)
24. MacKay, D.J.: A practical bayesian framework for backpropagation networks. Neural computation **4**(3), 448–472 (1992)
25. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (2007)
26. Neal, R.M.: Bayesian learning for neural networks, vol. 118. Springer Science & Business Media (2012)
27. Paisley, J., Blei, D., Jordan, M.: Variational bayesian inference with stochastic search. arXiv preprint arXiv:1206.6430 (2012)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), `http://www.aclweb.org/anthology/D14-1162`
29. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE **77**(2), 257–286 (1989)
30. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. arXiv preprint arXiv:1401.4082 (2014)
31. Šabata, T., Borovicka, T., Holena, M.: K-best viterbi semi-supervized active learning in sequence labelling. CEUR workshop proceedings pp. 144–152 (2017)
32. Sang, E.F., De Meulder, F.: Introduction to the conll-2003 shared task: Language-independent named entity recognition. arXiv preprint cs/0306050 (2003)
33. Scheffer, T., Decomain, C., Wrobel, S.: Active hidden markov models for information extraction. In: International Symposium on Intelligent Data Analysis. pp. 309–318. Springer (2001)
34. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing **45**(11), 2673–2681 (1997)
35. Seshadri, N., Sundberg, C.E.: List viterbi decoding algorithms with applications. IEEE transactions on communications **42**(234), 313–323 (1994)
36. Settles, B.: Active learning literature survey. Science **10**(3), 237–304 (1995)
37. Settles, B., Craven, M.: An analysis of active learning strategies for sequence labeling tasks. In: Proceedings of the conference on empirical methods in natural language processing. pp. 1070–1079. Association for Computational Linguistics (2008)
38. Shen, Y., Yun, H., Lipton, Z., Kronrod, Y., Anandkumar, A.: Deep active learning for named entity recognition. Proceedings of the 2nd Workshop on Representation Learning for NLP (2017). https://doi.org/10.18653/v1/w17-2630, `http://dx.doi.org/10.18653/v1/W17-2630`
39. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: International conference on machine learning. pp. 843–852 (2015)
40. Tomanek, K., Hahn, U.: Semi-supervised active learning for sequence labeling. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2. pp. 1039–1047. ACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA (2009)

Combating Stagnation in Reinforcement Learning Through 'Guided Learning' With 'Taught-Response Memory'*

Keith Tunstead[1][0000−0002−9769−1009] and Joeran Beel[1][0000−0002−4537−5573]

Trinity College Dublin, School of Computer Science and Statistics, Artificial Intelligence Discipline, ADAPT Centre, Dublin, Ireland
{tunstek,beelj}@tcd.ie

Abstract. We present the concept of Guided Learning, which outlines a framework that allows a Reinforcement Learning agent to effectively 'ask for help' as it encounters stagnation. Either a human or expert agent supervisor can then optionally 'guide' the agent as to how to progress beyond the point of stagnation. This guidance is encoded in a novel way using a separately trained neural network referred to as a 'Taught Response Memory' that can be recalled when another 'similar' situation arises in the future. This paper shows how Guided Learning is algorithm independent and can be applied in any Reinforcement Learning context. Our results achieved superior performance over the agents non-guided counterpart with minimal guidance, achieving, on average, increases of 136% and 112% in the rate of progression of the champion and average genomes respectively. This is due to the fact that Guided Learning allows the agent to exploit more information and thus, the agent's need for exploration is reduced.

Keywords: Active learning · Agent teaching · Evolutionary algorithms · Interactive adaptive learning · Stagnation

1 Introduction

One of the primary problems with training any kind of modern AI in a Reinforcement Learning environment is stagnation. Stagnation occurs when the agent ceases to make progress in solving the current task prior to either the goal or the agents maximum effectiveness being reached. The reduction of stagnation is an important topic for reducing training times and increasing overall performance in cases where training times are limited.

This paper will present a method to reduce stagnation and define a framework for a kind of interactive teaching/guidance where either a human or expert agent supervisor can guide a learning agent past stagnation.

* This publication emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number 13/RC/2106.

In terms of related work, we will briefly discuss Teaching and Interactive Adaptive Learning. The concept of Teaching[3] encompasses agent-to-agent [6], agent-to-human [8] and human-to-agent teaching [1]. Guided Learning is a form of Teaching that can take advantage of both human-to-agent and agent-to-agent. Interactive Adaptive Learning is defined as a combination of Active Learning, a type of Machine Learning where the algorithm is allowed to query some information source in order to obtain the desired outputs, and Adaptive Stream Mining which concerns itself with how the algorithm should adapt when dealing with time changing data [2].

2 Guided Learning

Guided Learning encodes guidance using what we refer to as Taught Response Memories (TRMs), which we define as: a memory of a series of actions that an agent has been taught in response to specific stimuli. A TRM is an abstract concept but its representation must allow for some plasticity in order to adapt the memory over time, this allows a TRM to tend towards a more optimal solution for a single stimulus or towards its applicability, more generally, to other stimuli. In this paper we represent TRMs as separately trained feed-forward neural networks. TRMs may consist of multiple actions and this can cause non-convergence when conflicting actions are presented, therefore we define a special case TRM, referred to as a Single Action TRM (SATRM). Using SATRMs, multiple actions can be split into their single action components, therefore removing any conflicting actions. Due their independence from the underlying algorithm, TRMs (and subsequently Guided Learning) can be used with any Reinforcement Learning algorithm.

The ideal implementation of Guided Learning can be best described using an example. In the game Super Mario Bros, when a reinforcement agent stagnates at the first green pipe (see Fig. 1 in Appendix A), the agent can request guidance from a supervisor. If no guidance is received within a given time period, the algorithm will continue as normal. Any guidance received is encoded as a new TRM. The TRM can be 'recalled' in order to attempt to jump over, not only the first green pipe but the second, and the third and so on. A TRM is 'recalled' if the current stimulus falls within a certain 'similarity threshold', $\theta < t$, of the stimulus for which the TRM was trained, i.e. $\theta = \arccos \frac{a.b}{|a||b|}$ where a and b are the stimulus vectors. Because each TRM is plastic, it can tend towards getting more optimal at either jumping over that one specific green pipe or jumping over multiple green pipes. This also helps in cases where guidance is sub-optimal. A full implementation of Guided Learning can recall the TRM, not only in the first level or in other levels of the game but in other games entirely with similar mechanics to the original game (i.e. another platform or 'jump and run' based game, where the agent is presented with a barrier in front of it). For more information please refer to the extended version of this manuscript [7].

3 Methodology

The effectiveness of a limited implementation of Guided Learning[1] will be measured using the first level of the game Super Mario Bros[2]. The underlying Reinforcement Learning algorithm used was Neural Evolution of Augmenting Topologies (NEAT)[5]. NEAT was chosen firstly due to it's applicability as a Reinforcement Learning algorithm and secondly due to NEATs nature as an Evolutionary Algorithm. The original intent was to reuse TRMs across multiple genomes. While this worked to an extent (see Avg Fitness metric in Fig. 3 in Appendix B.1), it was not as successful as originally hoped. This is because different genomes tend to progress in distinct ways and future work still remains in regards to TRM reuse. Stagnation was defined as evaluating 4 generations without the champion genome making progress.

To evaluate Guided Learning, a baseline was created that only consisted of the NEAT algorithm. The stimulus was represented as raw pixel data with some dimensionality reduction (see Fig. 2 in Appendix A). The Guided Learning implementation then takes the baseline and makes the following changes: 1) Allows the agent to 'ask for help' from a human supervisor when stagnation is encountered. 2) Encodes received guidance as SATRMs. 3) Activates SATRMs as 'similar' situations are encountered.

Both the baseline and Guided Learning algorithms were evaluated 50 times, each to the 150th generation. 'Best Fitness' and 'Average Fitness' results refer to the fitness of the champion genome and average fitness of the population at each generation respectively. Where 'fitness' is defined as the distance the agent moves across the level.

4 Results & Discussion

For Guided Learning, an average of 10 interventions were given over an average period of about 8 hours. Interventions were not given at each opportunity presented and were instead lazily applied, averaging to 1 intervention for every 3 requests. The run-time of Guided Learning was mostly hindered by the overhead of checking for stimulus similarity, this resulted in an extra run-time of about 2x the baseline. This run-time can be substantially improved with some future work.

Guided Learning achieved 136% and 112% improvements in the regression slopes for both the Mean Best Fitness and Mean Average Fitness respectively (see Fig. 3 in Appendix A). We also looked at the best and worst performing cases. These results can be seen in Fig. 4 and Table 2 in Appendix B.2.

[1] https://github.com/BeelGroup/Guided-Learning

[2] Disclaimer: The ROM used during the creation of this work was created as an archival backup from a genuine NES cartridge and was NOT downloaded/distributed over the internet.

The results obtained show good promise for Guided Learnings potential as such results were obtained with only a partial implementation and much future work still remains.

Some of the limitations of Guided Learning include the need for some kind of supervisor, its current run-time and its domain dependence i.e. a TRM for 'jump and run' games would not work in other games with different mechanics or reinforcement scenarios.

Future work will include: 1) Building Guided Learning using more state of the art Reinforcement Learning algorithms [4]. 2) Using a more generalized encoding of the stimulus to allow TRMs to be re-used more readily while still balancing the false-negative and false-positive activation trade-off (i.e. feeding raw pixel data into a trained classifier). 3) Implementing TRM adaptation. 4) Taking advantage of poorly performing TRMs as a method of showing the agent what *not* to do [3]. 5) Run-time optimization by offloading the similarity check and guidance request to separate threads, this would mean that the agent would no longer wait for input and TRM selection predictions can also be made as the current stimulus converges towards a valid TRM stimulus.

References

1. Hussein, A., Elyan, E., Gaber, M.M., Jayne, C.: Deep reward shaping from demonstrations. In: 2017 International Joint Conference on Neural Networks (IJCNN). pp. 510–517. IEEE (2017)

2. Kottke, D.: Interactive adaptive learning (2018), http://www.daniel.kottke.eu/2018/tutorial-interactive-adaptive-learning, [Online; accessed June 18, 2019]

3. Lin, L.J.: Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning **8**(3-4), 293–321 (1992)

4. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529 (2015)

5. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary computation **10**(2), 99–127 (2002)

6. Taylor, M.E., Carboni, N., Fachantidis, A., Vlahavas, I., Torrey, L.: Reinforcement learning agents providing advice in complex video games. Connection Science **26**(1), 45–63 (2014)

7. Tunstead, K., Beel, J.: Combating stagnation in reinforcement learning through 'guided learning' with 'taught-response memory' [extended version]. arXiv (2019)

8. Zhan, Y., Fachantidis, A., Vlahavas, I., Taylor, M.E.: Agents teaching humans in reinforcement learning tasks. In: Proceedings of the Adaptive and Learning Agents Workshop (AAMAS) (2014)

A Figures & Tables

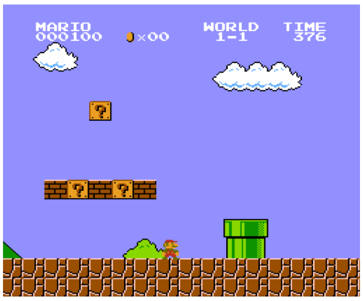
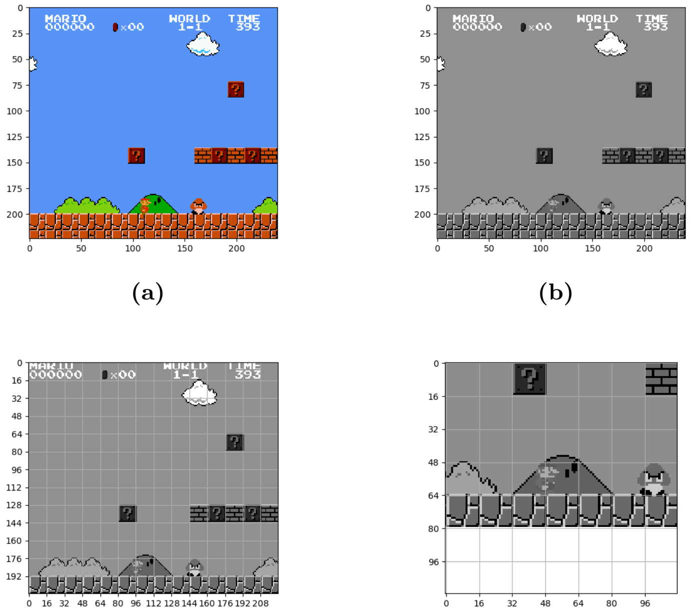


Fig. 1. First pipe encounter in Super Mario Bros.



(a)

(b)

(c)

(d)

Fig. 2. Input Reduction Pipeline Examples. **(a)** Raw RGB Frame **(b)** Grayscaled Frame **(c)** Aligned and Tiled Frame **(d)** Radius Tiles Surrounding Mario, $r = 4$

Table 1. NEAT Configuration Used During Evaluation

Parameter	**Value**
Initial Population Size	50
Activation Function	Sigmoid
Activation Mutation Rate	0
Initial Weight/Bias Distribution Mean	0
Initial Weight/Bias Distribution Std. Deviation	1
Weight & Bias Max Value	30
Weight & Bias Min Value	-30
Weight Mutation Rate	0.5
Bias Mutation Rate	0.1
Node Add Probability	0.2
Node Delete Probability	0.1
Connection Add Probability	0.3
Connection Delete Probability	0.1
Initial number of Hidden Nodes	6
Max Stagnation	20
Elitism	5

B Results Figures & Tables

B.1 Average Results Over 50 Trials

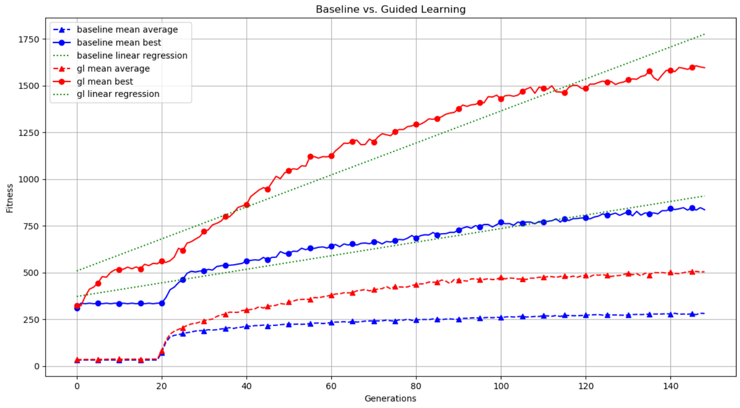
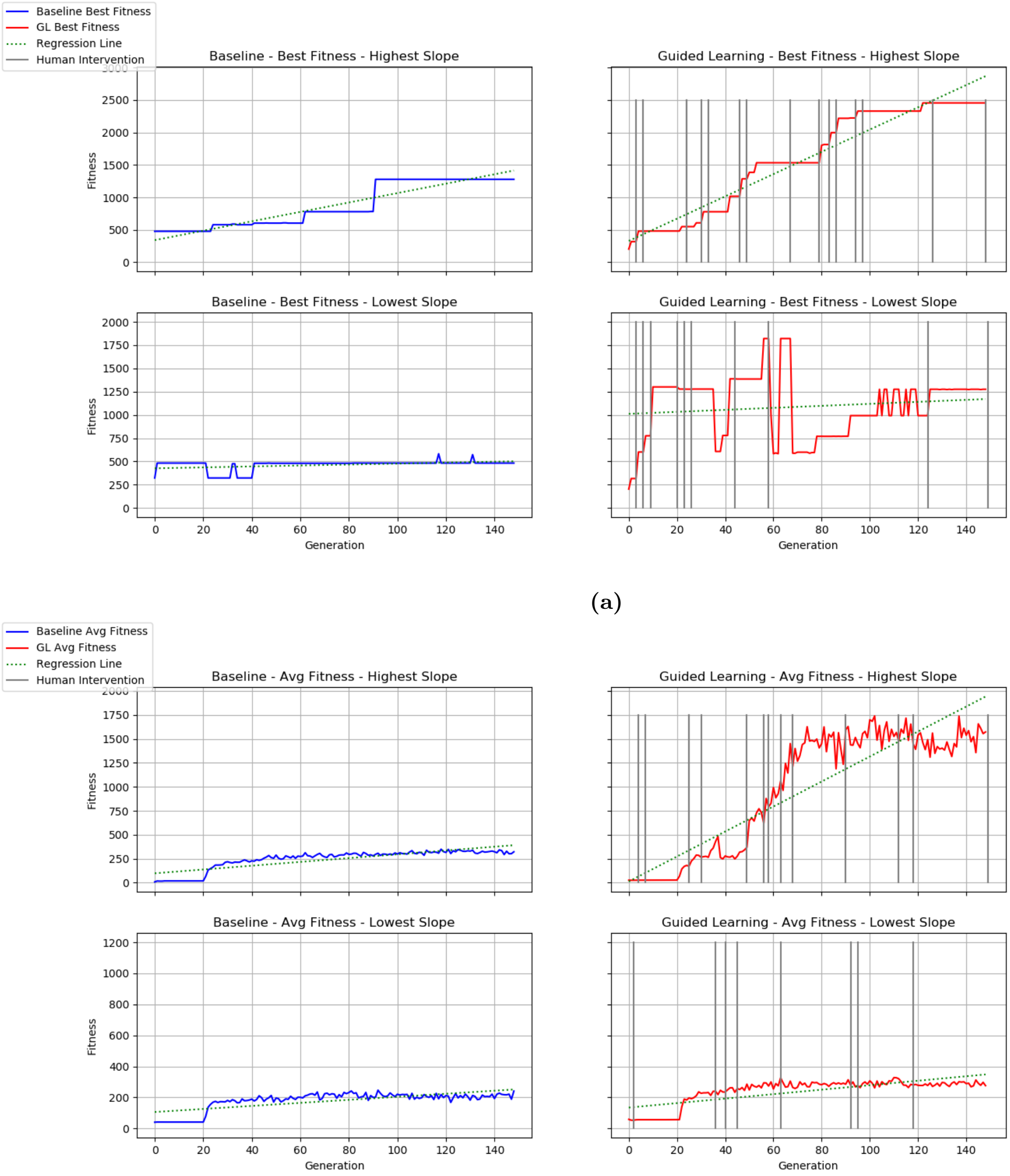


Fig. 3. Baseline vs. Guided Learning Average Results Per Generation (Higher is better).

B.2 Best & Worst Case Results



(a)

(b)

Fig. 4. Baseline vs. Guided Learning Best and Worst Case Results (Higher is better). **(a)** Best Fitness. **(b)** Avg Fitness.

Table 2. Baseline vs. Guided Learning Best and Worst Case Slope Results

	Baseline	**Guided Learning**	**Improvement**
Best Fitness (Highest Slope)	7.25	17.16	137%
Best Fitness (Lowest Slope)	0.51	1.07	110%
Avg Fitness (Highest Slope)	1.98	13.03	558%
Avg Fitness (Lowest Slope)	0.98	1.44	47%

Towards Active Simulation Data Mining*

Mirko Bunse[1], Amal Saadallah[1], and Katharina Morik[1]

TU Dortmund, AI Group, 44221 Dortmund, Germany
{firstname.lastname}@tu-dortmund.de

Abstract. Simulations have recently been considered as data generators for machine learning. However, the high computational cost associated with them requires a smart sampling of what to simulate. We distinguish between two scenarios of simulation data mining, which can be optimized with active learning and active class selection.

Keywords: Simulation · Active learning · Active class selection.

1 Introduction

Simulations are powerful tools for investigating the behavior of complex systems in science and engineering. Recently, there is an increase of attention towards the employment of simulated data in machine learning, an integration that is sometimes termed *simulation data mining* [11,2,4,12]. Its applications range from integrated circuit design [13] over milling processes [9], mechanized tunneling [8], robotized surgery [7], and cancer treatment [5] to astro-particle physics [3].

The goal of simulation data mining is to reason about a real system under study by learning from data which is generated by a simulation of that system. The benefit of this paradigm is that less or even no data is required from the actual system. Acquiring "real" data would often be costly or even be infeasible, e.g. if the actual system is still in the design phase and not yet deployed. Oppositely, simulations have the potential to provide large volumes of data, only at the expense of their computation. However, the need for accurate simulations often leads to complex simulation models (e.g. 3D numerical Finite-Element simulations), which result in high costs associated with data generation. The time and computational resources required by simulations motivate the active sampling of data, more precisely active learning (AL) [10] and active class selection (ACS) [6]. Both of these frameworks seek to select the minimal amount of training data while maximizing the performance of a prediction model trained with that data. In this short paper, we argue that there are two different strategies for the simulation of training data which distinctively correspond to AL and ACS. In fact, a simulation may either generate labels from a set of input features [11,12,13,2,9,8] or it may generate feature vectors from input labels [7,1]. The need for cost efficiency thus makes simulation data mining an imminent application scenario for methods from AL and from ACS.

* This work has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876. "Providing Information by Resource-Constrained Data Analysis", projects C3 and B3. http://sfb876.tu-dortmund.de

2 Active Sampling from Simulated Data

Every simulation is based on some kind of generative model. Such a *simulation model* may comprise analytical, geometric, agent-based, and probabilistic modeling approaches which represent the dynamics of the studied system. Namely, such a model represents how the state $\boldsymbol{s} \in \mathcal{S}$ of the system evolves over time:

$$Sim_{\boldsymbol{\rho}}(\boldsymbol{s}_t, \Delta t) \ = \ \boldsymbol{s}_{t+\Delta t}\,, \quad 0 \leq t \leq T, \tag{1}$$

where $\boldsymbol{\rho} \in \mathcal{P}$ is a vector of simulation parameters, which can be directly related to the parameters of the real system or process. In this view, the simulation is a fixed black box which encodes domain knowledge up to minor details. In the following, we distinguish between two scenarios in which machine learning models are trained on simulated data.

2.1 Forward Learning Scenario

In the first learning scenario, the simulation model has the same direction of inference as the machine learning model $f : \mathcal{X} \to \mathcal{Y}$ that is to be trained. This means that the initial state $\boldsymbol{s}_0 \in \mathcal{S}$ of the simulation is a function of the feature vector $\boldsymbol{x} \in \mathcal{X}$. The simulation then comprises multiple steps $\boldsymbol{s}_1 \in \mathcal{S}, \boldsymbol{s}_2 \in \mathcal{S}, \ldots$ until a label $y \in \mathcal{Y}$ is obtained in the the final state $\boldsymbol{s}_T \in \mathcal{S}$. Thus, the simulation and the machine learning model both infer y from $\boldsymbol{x}$, as illustrated in Fig. 1. This learning scenario is probably the most common to date, being approached for example in [11,12,13,2,9,8].

$$\begin{tikzcd}
 & \boldsymbol{s}_0 \arrow[r, dashed, "Sim_{\boldsymbol{\rho}}"] & \boldsymbol{s}_1 \arrow[r, dashed, "Sim_{\boldsymbol{\rho}}"] & \cdots \arrow[r, dashed, "Sim_{\boldsymbol{\rho}}"] & \boldsymbol{s}_T \arrow[d, dashed] \\
 {} \arrow[r, dashed] & \boldsymbol{x} \arrow[u, dashed] \arrow[rrr, "f"'] & & & y
\end{tikzcd}$$

Fig. 1. In the *forward* scenario, the prediction model $f : \mathcal{X} \to \mathcal{Y}$ has the same direction of inference as the simulation $Sim_{\boldsymbol{\rho}}$ from Eq. 1.

Since the mappings from $\boldsymbol{x}$ to $\boldsymbol{s}_0$ and from $\boldsymbol{s}_T$ to y are given by the problem statement, we could use the simulation to predict y directly—without learning another model f from simulated data. However, simulations often encompass even those details of the analyzed system that are only minor for the prediction task at hand. The computational resources required to compute data from such a precise model limit the resource efficiency of the simulation with respect to the prediction task. It is therefore often not feasible to run a simulation for prediction, particularly for resource-aware or real-time applications. Machine learning can then be used to build surrogate models which solve the prediction task efficiently [9,8]. The simulation can take the role of an oracle $o_{\mathrm{AL}} : \mathcal{X} \to \mathcal{Y}$, so that an AL technique can optimize the data being simulated.

2.2 Backward Learning Scenario

In the second scenario, the goal is to learn a prediction model of the “opposite direction” of the simulation. In other words, the prediction task to find the causes of observed effects. This task is modeled by the label y defining the input of the simulation and a corresponding feature vector $\boldsymbol{x}$ being produced, as outlined in Fig. 2. Since the machine learning model now solves another task than the simulation, it is able to achieve analysis goals which can not be achieved with the simulation alone. This second scenario is applied, for example, in robotized surgery, where the force which caused a deformation is predicted [7], or in astro-particle physics, where particle properties are predicted from indirect observations [1,3].

$$\begin{tikzcd}
& \boldsymbol{s}_0 \arrow[r, dashed, "Sim_{\boldsymbol{\rho}}"] & \boldsymbol{s}_1 \arrow[r, dashed, "Sim_{\boldsymbol{\rho}}"] & \cdots \arrow[r, dashed, "Sim_{\boldsymbol{\rho}}"] & \boldsymbol{s}_T \arrow[d, dashed] \\
{} \arrow[r, dashed] & y \arrow[u, dashed] & & & \boldsymbol{x} \arrow[lll, "f"']
\end{tikzcd}$$

Fig. 2. In the *backward* scenario, the goal is to predict causes of observed effects. Thus, the direction of infence differs between $Sim_{\boldsymbol{\rho}}$ and f.

Other than in the forward scenario, a “backward” simulation can not predict y from $\boldsymbol{x}$. It can thus not be used as an AL oracle. However, we can use the simulation as the data generator $o_{\text{ACS}} : \mathcal{Y} \rightarrow \mathcal{X}$ that is assumed by ACS. One reason for distinguishing the two scenarios is thus the applicability of active sampling techniques. AL is only amenable in the forward scenario, ACS only in the backward case.

2.3 Active Sampling with Simulation Parameters

The goal of AL and ACS is to reduce the cost of training data generation. Starting from an initial data set, the simulation candidates are scored according to a selection criterion s and the best candidates are being simulated until a stopping criterion is met after some iterations. In this framework, AL scores feature vectors and ACS—in contrast—scores labels.

$$\begin{aligned} s_{\text{AL}} &: \mathcal{X} \rightarrow \mathbb{R} \\ s_{\text{ACS}} &: \mathcal{Y} \rightarrow \mathbb{R} \end{aligned}$$

Having a simulation, we can generalize this concept to a scoring of all simulation inputs, also comprising the auxiliary simulation parameters $\boldsymbol{\rho} \in \mathcal{P}$. Namely, AL can score each $(\boldsymbol{x}, \boldsymbol{\rho})$ and ACS can score each $(y, \boldsymbol{\rho})$ to have a higher chance of identifying the relevant input sub-spaces and to improve efficiency further.

3 Conclusion

We distinguish between two scenarios in which machine learning models are trained from simulated data. Our distinction corresponds to the applicability of AL and ACS, a property not previously detailed in simulation data science. Moreover, we conceive that active sampling techniques can be improved by accounting for the parameters of the simulation.

In upcoming work, we will further elaborate the paradigm of learning from simulations. In this regard, we deem data quality a particular issue because simulated data does not always picture the real system exactly. This problem may be tackled with transfer learning or domain adaptation techniques, which make the differences between multiple data sources—the simulation and the real system—explicit. Therefore, we consider simulation data science a promising use case also for combinations of active sampling and transfer learning.

References

1. Bockermann, C., Brügge, K., Buss, J., Egorov, A., Morik, K., Rhode, W., Ruhe, T.: Online analysis of high-volume data streams in astroparticle physics. In: Proc. of the ECML-PKDD, Part III. LNCS, vol. 9286, pp. 100–115. Springer (2015)
2. Brady, T.F., Yellig, E.: Simulation data mining: a new form of computer simulation output. In: Proc. of the 37th Winter Simulation Conf. pp. 285–289. IEEE (2005)
3. Bunse, M., Piatkowski, N., Morik, K., Ruhe, T., Rhode, W.: Unification of deconvolution algorithms for Cherenkov astronomy. In: Proc. of the 5th Int. Conf. on Data Science and Advanced Analytics (DSAA). pp. 21–30. IEEE (2018)
4. Burrows, S., Stein, B., Frochte, J., Wiesner, D., Müller, K.: Simulation data mining for supporting bridge design. In: Proc. of the 9th Australasian Data Mining Conf. (AusDM). CRPIT, vol. 121, pp. 163–170. Australian Computer Society (2011)
5. Deist, T., Patti, A., Wang, Z., Krane, D., Sorenson, T., Craft, D.: Simulation assisted machine learning (2018), http://arxiv.org/abs/1802.05688, under review
6. Lomasky, R., Brodley, C.E., Aernecke, M., Walt, D., Friedl, M.A.: Active class selection. In: Proc. of the ECML. LNCS, vol. 4701, pp. 640–647. Springer (2007)
7. Mendizabal, A., Fountoukidou, T., Hermann, J., Sznitman, R., Cotin, S.: A combined simulation and machine learning approach for image-based force classification during robotized intravitreal injections. In: Proc. of the 21st Int. Conf. on Medical Image Computing and Computer Assisted Intervention (MICCAI). LNCS, vol. 11073, pp. 12–20. Springer (2018)
8. Saadallah, A., Alexey, E., Cao, B.T., Freitag, S., Morik, K., Meschke, G.: Active learning for accurate settlement prediction using numerical simulations in mechanized tunneling. Procedia CIRP (2019)
9. Saadallah, A., Finkeldey, F., Morik, K., Wiederkehr, P.: Stability prediction in milling processes using a simulation-based machine learning approach. In: 51st CIRP Conf. on Manufacturing Systems. Elsevier (2018)
10. Settles, B.: Active Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers (2012)
11. Shao, Y., Liu, Y., Ye, X., Zhang, S.: A machine learning based global simulation data mining approach for efficient design changes. Advances in Engineering Software **124**, 22–41 (2018)
12. Trittenbach, H., Gauch, M., Böhm, K., Schulz, K.: Towards simulation-data science – a case study on material failures. In: Proc. of the 5th Int. Conf. on Data Science and Advanced Analytics (DSAA). pp. 450–459. IEEE (2018)
13. Wang, L., Marek-Sadowska, M.: Machine learning in simulation-based analysis. In: Proc. of the Int. Symp. on Physical Design (ISPD). pp. 57–64. ACM (2015)

Active Feature Acquisition for Opinion Stream Classification under Drift

Ranjith Shivakumaraswamy$^{1[0000-0003-0868-4016]}$, Christian Beyer$^{1[0000-0001-8604-9523]}$, Vishnu Unnikrishnan$^{1[0000-0002-0086-594X]}$, Eirini Ntoutsi$^{2[0000-0001-5729-1003]}$, and Myra Spiliopoulou$^{1[0000-0002-1828-5759]}$

[1] Otto-von-Guericke-University Magdeburg, Germany
ranjiths1492@gmail.com, {christian.beyer,vishnu.unnikrishnan,myra}@ovgu.de
[2] Leibniz University Hannover, Germany
ntoutsi@kbs.uni-hannover.de

Abstract. Active stream learning is frequently used to acquire labels for instances and less frequently to determine which features should be considered as the stream evolves. We introduce a framework for active feature selection, intended to adapt the feature space of a polarity learner over a stream of opinionated documents. We report on the first results of our framework on substreams of reviews on different product categories.

Keywords: Active Feature Acquisition · Opinion Stream Classification

1 Introduction

Opinion stream classification algorithms assign a polarity label to each arriving opinionated document. The feature space over the stream may change though, e.g. when new product appear and the words/phrasing used by customers who reviewed them changes. Feature space adaption can benefit from an active learning approach, where a human expert specifies the features of importance.

Contardo et al. [5] use reinforcement learning to acquire features, and also consider feature acquisition cost. Huang et al. [8] take uncertainty into account. The "sequential feature acquisition framework" of Shim et al. [12] acquires one feature at a time until the desired model confidence is achieved. These approaches are for static data, though, which are processed in their entirety to build the model. In the stream context, Barddal et al. [2] survey methods that detect feature drift and select features for learning, under the assumption that all features are known in advance. We do not make this assumption. Rather, whenever drift is detected, we use words from recent documents and rebuild the feature space.

We propose a framework for active feature selection on a stream. It consists of: an active learner of features (ALF) that ranks features on importance; a recommender (RALF) that invokes ALF and then recommends a feature subspace to be replaced with the new features; a drift monitor that invokes RALF when model quality decreases. In the next section we present our framework. Section 3 contains our first results. Section 4 concludes our study.

2 Workflow Over the Document Stream

Our framework slides a window W of n epochs (here: weeks) over the stream, learning on n epochs and testing on the epoch $n + 1$.

Module ALF for Feature Ranking: Our active feature selector ALF ranks features on importance. Feature ranking methods include mutual information, information gain, document frequency thresholding, chi-square and document frequency thresholding (DFT) as discussed by Basu et al [3], Distinguishing Feature Selector (DFS), Odds Ratio and Normalized Difference Measure (NDM) as studied in [1], Gini-index, signed chi-square and signed information gain [10], the stratified feature ranking method of [4] and the approach proposed by [6]. We opted for the Distinguishing Feature Selector (ALF-DFS) and the Gini (ALF-Gini) because they were found to have the most competitive performance [14].

Module RALF for Feature Subspace Recommendation: The recommender takes as input the size M of the subspace to be replaced and invokes ALF for feature ranking. Currently we use $M = \frac{FeatureSpaceSize}{2}$. We have four variants of RALF:

- *Baseline:* invokes ALF-Gini on the data inside the current window
- *Oracle-Random:* picks randomly M features from the feature space of the *next* epoch (the epoch $n + 1$, i.e. the first epoch in the future)
- *Oracle-Gini:* invokes ALF-Gini on epoch n+1 and returns the top-M features
- *Oracle-DFS:* similar to ALF-Gini, but invokes ALF-DFS on epoch $n + 1$

Hence, the Oracle variants simulate an expert who knows which features will become important in the immediate future. We use the top-M of these features to replace the *least important ones* of the current feature space, thus preserving the presently informative features still.

Stream Classification Core: The opinion stream learner replaces the least informative features (according to ALF's ranking) with the features suggested by RALF. It re-learns on the current window and uses the next epoch for testing. Then, the window shifts by one epoch, forgetting the least recent one.

Drift-driven Feature Space Update: Drift monitor that invokes RALF if and only if drift occurs. For drift detection we use the method of Gama et al [7].

3 Experiments and Results

We compared the RALF variants to a default model that does not change the feature space. We performed prequential evaluation, aggregated the SGD log loss values every two months. We used Friedman test with Iman-Davenport modification, rejecting the H_0 for p-values ≤ 0.01, and then applied Nemenyi post-hoc test. All experiments and results are in [13].

Data Setup: We use the “clothing, shoes and jewelry” reviews (substream C), “health and personal care” (substream H) and “sports and outdoors” (S) from the Amazon data set of [9] (`http://jmcauley.ucsd.edu/data/amazon/`), from 01/2011 to 01/2013. There were very few reviews before 2011 and a steep increase of positive ones from 2013 on: this product-independent drift calls for conventional classifier adaption, which is beyond our scope. We map ratings 1 and 2 to “Negative”, 4 and 5 to “Positive”, and 3 to “Neutral”.

Feature Drift Imputation: We start and stop the substream of each product category at specific time points (see Fig. 1). Hence, product-specific words appear only at given time intervals. We slide a window of 5 weeks in one-week steps over this stream. We build an initial model from the first three weeks, i.e. only from substream C. The first drift occurs when substream H starts.

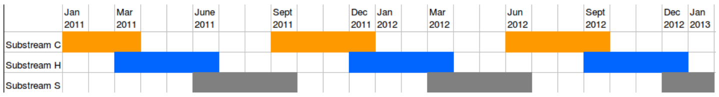


Fig. 1. One substream per product category, shifted over time to simulate feature drift

Setup of the Components: As classification core we use Stochastic Gradient Descent (SGD) of scikit-learn (alpha = 0.001, l2 penalty and hinge loss). For text preparation, we use the components of [11]. We build the feature space using bag-of-words (“words”: 3-grams) and TFIDF, and invoke the dictionary vectorizer of scikit-learn. We vary the feature space size $M_{full} = 500, 1000, 5000, 10000, 15000$, so RALF replaces the $M = M_{full}/2$ least important features.

Results: The default model always had inferior performance. Hence updating the feature space is beneficial as response to drift caused through the introduction of new products.Oracle-DFS performed best. Oracle-Gini was within the critical distance to it. Oracle-Random improved as the feature space size increased.

The Baseline, which uses ALF-Gini without benefiting from an Oracle, is comparable to Oracle-Gini and Oracle-Random, It is better than the default model except for $M_{full} = 500$ (where it is within the critical distance from the default model). Hence, ALF-Gini can improve model performance by replacing the least informative features in the *current* window, when feature drift occurs.

4 Conclusions

We presented an active feature selection framework for a stream of opinionated documents. Upon drift detection, our framework re-ranks the features with help of the Oracle and replaces the least informative old features with the most informative new ones. We evaluated our framework by simulating topic drift. We found that replacing a feature subspace in the presence of drift is beneficial, even if there is no Oracle. We next plan to vary the size and position of the feature subspace to be replaced. Replacing the currently most informative features instead of the least informative ones might be better under concept shift.

Acknowledgement

This work is partially funded by the German Research Foundation, project OSCAR "Opinion Stream Classification with Ensembles and Active Learners". We further thank Elson Serrao who made the basic components of opinion stream mining available under `https://github.com/elrasp/osm`.

References

1. Asim, M.N., Wasim, M., Ali, M.S., Rehman, A.: Comparison of feature selection methods in text classification on highly skewed datasets. In: 1st Int. Conf. on Latest Trends in Electrical Engineering and Computing Technologies (INTELLECT). pp. 1–8. IEEE (2017)
2. Barddal, J.P., Gomes, H.M., Enembreck, F., Pfahringer, B.: A survey on feature drift adaptation: Definition, benchmark, challenges and future directions. Journal of Systems and Software **127**, 278–294 (2017)
3. Basu, T., Murthy, C.: Effective text classification by a supervised feature selection approach. In: 12th IEEE ICDM, Workshops Volume. pp. 918–925. IEEE (2012)
4. Chen, R., Sun, N., Chen, X., Yang, M., Wu, Q.: Supervised feature selection with a stratified feature weighting method. IEEE Access **6**, 15087–15098 (2018)
5. Contardo, G., Denoyer, L., Artières, T.: Sequential cost-sensitive feature acquisition. In: Int. Symp. on Intelligent Data Analysis. pp. 284–294. Springer (2016)
6. Fattah, M.A.: A novel statistical feature selection approach for text categorization. Journal of Information Processing Systems **13**(5) (2017)
7. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Brazilian Symposium on Artificial Intelligence. pp. 286–295. Springer (2004)
8. Huang, S.J., Xu, M., Xie, M.K., Sugiyama, M., Niu, G., Chen, S.: Active feature acquisition with supervised matrix completion. In: 24th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining. pp. 1571–1579. ACM (2018)
9. McAuley, J., Targett, C., Shi, Q., Van Den Hengel, A.: Image-based recommendations on styles and substitutes. In: 38th Int ACM SIGIR Conf on Research and Development in Information Retrieval. pp. 43–52. ACM (2015)
10. Ogura, H., Amano, H., Kondo, M.: Comparison of metrics for feature selection in imbalanced text classification. Expert Systems with Applications **38**(5), 4978–4989 (2011)
11. Serrao, E., Spiliopoulou, M.: Active stream learning with an oracle of unknown availability for sentiment prediction. In: IAL@ECML PKDD. pp. 36–47 (2018)
12. Shim, H., Hwang, S.J., Yang, E.: Joint active feature acquisition and classification with variable-size set encoding. In: Advances in Neural Information Processing Systems. pp. 1368–1378 (2018)
13. Shivakumaraswamy, R.: Active learning over text streams. Tech. rep., Otto-von-Guericke-University Magdeburg Department of Computer Science (2019)
14. Uysal, A.K.: An improved global feature selection scheme for text classification. Expert systems with Applications **43**, 82–92 (2016)