Daniel Kottke Vincent Lemaire Adrian Calma
Georg Krempl Andreas Holzinger

IAL@ECML PKDD 2020

Workshop on Interactive Adaptive Learning

Proceedings

The European Conference on Machine Learning and
Principles and Practice of Knowledge Discovery in Databases
(ECML PKDD 2020)

Ghent, Belgium, September 14, 2020 (virtual conference)

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 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 and Tutorial), IJCNN 2018 in Rio (Tutorial), ECML PKDD 2018 in Dublin (Workshop), and ECML PKDD 2019 in Würzburg (Workshop and Tutorial).
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. Our objective is to bridge the communities researching and developing these techniques and systems in machine learning and data mining. Therefore, we welcome contributions that present a novel problem setting, propose a novel approach, or report experience with the practical deployment of such a system and raise unsolved questions to the research community.

All in all, we accepted four regular papers (4 papers submitted) and four short papers (6 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 2020

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

Organization

Organizing Committee

Adrian Calma	
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
Alexis Bondu	Orange Labs
Klemens Böhm	Karlsruhe Institute of Technology
Martin Holena	Institute of Computer Science
Dino Ienco	IRSTEA
George Kachergis	
Edwin Lughofer	Johannes Kepler University Linz
Shreyasi Pathak	University of Twente
Ingo Scholtes	University of Zurich
Carlos Soares	LIAAD-INESCTEC, Porto
Stefano Teso	Katholieke Universiteit Leuven
Holger Trittenbach	Karlsruhe Institute of Technology
Sebastian Tschiatschek	

Table of Contents

Invited Talks

When Humans and AI Collide 1
Kori Inkpen

The Rapid Growth of Human-in-the-Loop Machine Learning 2
Robert Munro

How to Measure Uncertainty in Uncertainty Sampling for Active Learning 3
Eyke Hüllermeier

From Explainable AI to Human-Centered AI 4
Andreas Holzinger

Full Research Papers

Improving Unsupervised Domain Adaptation with Representative Selection Techniques 5
I-Ting Chen and Hsuan-Tien Lin

On the Transferability of Deep Neural Networks for Recommender System 22
Duc Nguyen, Hao Niu, Kei Yonekawa, Mori Kurokawa, Chihiro Ono, Daichi Amagata, Takuya Maekawa and Takahiro Hara

Learning Active Learning at the Crossroads? Evaluation and Discussion . 38
Louis Desreumaux and Vincent Lemaire

The Effects of Reluctant and Fallible Users in Interactive Online Machine Learning 55
Agnes Tegen, Paul Davidsson and Jan A. Persson

Short Papers

Active Learning for LSTM-autoencoder-based Anomaly Detection in Electrocardiogram Readings 72
Tomáš Šabata and Martin Holena

Towards Landscape Analysis in Adaptive Learning of Surrogate Models . . 78
Zbyněk Pitra and Martin Holena

VIAL-AD: Visual Interactive Labelling for Anomaly Detection - An approach and open research questions 84
Andreas Theissler, Anna-Lena Kraft, Max Rudeck and Fabian Erlenbusch

Get a Human-In-The-Loop: Feature Engineering via Interactive Visualizations 90
Dimitra Gkorou, Maialen Larranaga, Alexander Ypma, Faegheh Hasibi and Robert Jan van Wijk

When Humans and AI Collide

Kori Inkpen

Microsoft Research, USA
kori@microsoft.com

Abstract. As the use of AI in society grows and evolves, we see both opportunities and risks for these technologies. While AI has already shown strong performance in some areas, there are still many domains where the potential impact of AI will depend on the interaction between Humans and AI. So what happens when Humans and AI disagree? Who do you trust? And what happens when the Human, the AI, or both are biased? We need to continue to evolve our understanding of how humans and AI systems can work together, effectively harnessing the benefits of both systems, and mitigating their inherent biases. This talk will share results from our work on Human-AI complementarity, and the intersection of Human and AI bias.

The rapid growth of Human-in-the-Loop Machine Learning

Robert Munro

Machine Learning Consulting, USA
robert.munro@gmail.com

Abstract. In the last few years Human-in-the-Loop Machine Learning has quickly become the dominant paradigm in many industries adopting AI for the first time. More than 90% of machine learning applications today are powered by supervised machine learning, including autonomous vehicles, in-home devices, and every item you purchase on-line. There are thousands of professional annotators and subject matter experts fine-tuning each underlying model by annotating new data. This talk will highlight different use cases in Human-in-the-Loop Machine Learning in industries including finance, healthcare, entertainment, retail, and disaster response.

How to measure uncertainty in Uncertainty Sampling for Active Learning

Eyke Hüllermeier

University Potsdam, Germany
eyke@uni-paderborn.de

Abstract. Various strategies for active learning have been proposed in the machine learning literature. In uncertainty sampling, which is among the most popular approaches, the active learner sequentially queries the label of those instances for which its current prediction is maximally uncertain. The predictions as well as the measures used to quantify the degree of uncertainty, such as entropy, are traditionally of a probabilistic nature. Yet, alternative approaches to capturing uncertainty in machine learning, alongside with corresponding uncertainty measures, have been proposed in recent years. In particular, some of these measures seek to distinguish different sources and to separate different types of uncertainty, such as the reducible (epistemic) and the irreducible (aleatoric) part of the total uncertainty in a prediction. This talk elaborates on the usefulness of such measures for uncertainty sampling and compares their performance in active learning. To this end, uncertainty sampling is instantiated with different measures, the properties of the sampling strategies thus obtained are analyzed and compared in an experimental study.

From Explainable AI to Human-Centered AI

Andreas Holzinger

Medical University Graz, Austria
andreas.holzinger@human-centered.ai

Abstract. The problem of explainability is as old as AI itself and classic AI represented comprehensible retraceable approaches. Their weakness was in dealing with non-linearities and the intrinsic uncertainties of medical data. Advances in data-driven statistical machine learning have led to the current renaissance of AI, but the solutions are becoming increasingly complex and opaque. Due to increasing social, ethical, and legal aspects of AI in medicine, explainable AI (xAI) is attracting much interest within the international research community. While xAI deals with the implementation of transparency and traceability of statistical black-box machine learning methods, there is a pressing need to go beyond xAI, e.g. to extent explainability with causability. The integrative backbone for this approach is in interactive machine learning with the human-in-the-loop because a human domain expert complements AI with implicit knowledge. Humans are robust, can generalize from few examples, understand relevant representations and concepts and are able to explain causal links between them. Consequently, more research is needed on how human experts explain their decisions by examining their strategies, as they are (but not always) able to describe the underlying explanatory factors. Formalized, these can be used to build structural causal models of human decision making and characteristics can be mapped back to train AI. Finally, such an AI-ecosystem needs advanced Human-AI interfaces, that allow to ask questions of why, but also to ask for counterfactuals, i.e. what-if. This interactivity between human and AI will contribute to enhance robustness, reliability, accountability, fairness and trust in AI and foster ethical responsible machine learning with the human-in-control.

Improving Unsupervised Domain Adaptation with Representative Selection Techniques

I-Ting Chen and Hsuan-Tien Lin

Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
{r06922136,htlin}@csie.ntu.edu.tw

Abstract. Domain adaptation is a technique that tackles the dataset shift scenario, where the training (source) data and the test (target) data can come from different distributions. Current research works mainly focus on either the covariate shift or the label shift settings, each making a different assumption on how the source and target data are related. Nevertheless, we observe that neither of the settings can perfectly match the needs of a real-world bio-chemistry application. We carefully study the difficulties encountered by those settings on the application and propose a novel method that takes both settings into account to improve the performance on the application. The key idea of our proposed method is to select examples from the source data that are similar to the target distribution of interest. We further explore two selection schemes, the hard-selection scheme that plugs similarity into a nearest-neighbor style approach, and the soft-selection scheme that enforces similarity by soft constraints. Experiments demonstrate that our proposed method not only achieves better accuracy for the bio-chemistry application but also shows promising performance on other domain adaptation tasks when the similarity can be concretely defined.

Keywords: Domain Adaptation, Dataset Shift, Covariate Shift, Label Shift.

1 Introduction

Machine learning has been a high-profile topic and succeeded in various kinds of real-world tasks due to the vast amount of labeled data. However, collecting well-labeled data from scratch is time and labor-consuming. Therefore, in many applications [21], we hope that the model trained on one task could generalize to another related task. For example, consider an object recognition task that tries to distinguish ten different products based on their images on e-commerce websites. It is relatively easy to crawl and gather well-labeled data from the websites to train a classifier. After training the classifier, we may encounter another task where we hope that the users can easily recognize a product by taking pictures with their smartphones. Given that it is harder to gather well-labeled data from the users to train a classifier, we hope to reuse the data and/or the classifier obtained in the former task to tackle the latter one. Owing to the differences in brightness, in angle, and in picture quality between images taken from the two tasks, the same-distribution assumption on the training and test data may not hold. This scenario is called dataset shift [17], where the training(source) and test(target) data can come from different distributions.

A family of techniques that aim at tackling the dataset shift problem is domain adaptation (DA). In this work, we try to solve the more challenging unsupervised domain adaptation (UDA) problem, where we can only access the labeled source data and unlabeled target data in the training phase. The goal of UDA is to learn a model from these data and to achieve good performance on the target domain. Intuitively, learning under UDA is not possible if the source and target domains do not share any properties. Previous works on UDA thus make assumptions about the properties shared by the two domains and design algorithms based on the assumptions. Two major assumptions, covariate and label shift, have been considered separately in previous research works.

The assumption of covariate shift considers the mismatch of feature distribution between the source and target domains. Furthermore, it is assumed that the labels of both domains are drawn from the same conditional distribution given the features. There are two main families of methods designed under this assumption, namely, the re-weighting method [8, 20, 25], and the adversarial training method [3, 12, 13, 19]. They solve the same problem from different perspectives: Re-weighting based method estimates the difference in feature distributions between the source and target domains, whereas an adversarial training method aligns those distributions directly. The label shift assumption refers to the change of label distributions between the source and target domain while assuming that the features of both domains are drawn from the same conditional distribution given the label. Previous works focus on utilizing re-weighting [1, 11, 26] to solve this task. They estimate the difference between source and target domain label distributions.

Most recent works extend from the two settings and demonstrate promising performance. However, motivated by a real-world bio-chemistry application, we find that current domain adaptation methods designed for only one of the two assumptions cannot cope with all the application needs. We carefully examine the application and find it comes with the shift of label distribution that can be easily observed from the polarity of label distribution. However, the assumption that the conditional distribution given label does not seem to be the same, violating label shift assumption. Accordingly, we must use covariate shift assumption to model this application. Here comes the problem: If the application is tackled with the covariate shift assumption using adversarial training, the label distribution should be the same on the aligned data, violating the polarity property of the dataset. Therefore, we conclude that this application requires considering *both* the covariate shift and label shift properly. In this paper, we study how to follow the covariate shift assumption while taking the possible label shift into account for the bio-chemistry application. [24] also tries to tackle the same issue. They use adversarial training while imposing the constraint on the model. Therefore, the model would not perfectly align the distribution of source and target data.

Inspired by some intuitive toy examples, we find that selecting representative examples from the source data allows us to construct a similar-feature and similar-label subset of the source data that resolves both covariate shift and label shift. If the feature space implicitly encodes the distance between two features with physical meaning, we can construct the subset through the nearest-neighbor algorithm by considering the distance as the similarity measure. Based on this finding, we propose two methods, Hard/Soft Distance-Based Selection, to handle different situations. The hard selection directly uses the subset of the source data we construct to train the model, whereas the soft selection enforces similarity on the subset by adding a soft constraint.

Experiments show that our methods successfully capture the structural information and utilize the distance-based similarity and thus mitigate the impact from the label shift in the application. To test the performance of our methods in high-dimension space (e.g., image space), we also do experiments on the benchmark dataset (digits). Further, we extend our methods to tackle this scenario and have promising experimental results. Finally, we discuss what are the good situations to utilize our methods, through a simple noisy source data experiment.

Our contributions of this thesis include

1. We carefully study the difficulties encountered by concurrent UDA methods on a real-world application.
2. We propose two methods based on representative selection to overcome the difficulties.
3. We study how the proposed methods can be extended in different scenarios.

2 Background

2.1 Notation and Problem Setup

We consider a K-way classification task and let X and Y represent the random variables for the feature and label respectively, where $Y = \{0, \ldots, K-1\}$. We denote the joint distributions for the source and target domains as $P_S(X, Y)$ and $P_T(X, Y)$. The marginal distributions of X and Y in the source domain are defined as $P_S(X)$ and $P_S(Y)$. Similarly, $P_T(X)$ and $P_T(Y)$ represent the marginal distributions of X and Y in the target domain. The conditional label distributions in the two domains are denoted by $P_S(Y|X)$, $P_T(Y|X)$. $P_S(X|Y)$ and $P_T(X|Y)$ stand for the conditional feature distribution in the two domains.

We consider the UDA setting in this thesis. There exists a set of labeled data $\mathcal{D}_S = \{(x_i, y_i)\}_{i=1}^{n}$ in the source domain, where each instance (x_i, y_i) is drawn i.i.d. from $P_S(X, Y)$. In the target domain, we have only a set of unlabeled data $\mathcal{D}_T = \{\tilde{x}_j\}_{j=1}^{m}$, where each instance $\tilde{x}_j$ is drawn i.i.d. from $P_T(X)$.

Our goal is to train a classifier $f\colon X \to Y$, based on $\mathcal{D}_S$ and $\mathcal{D}_T$ and then predict the corresponding labels of $\mathcal{D}_T$. Note that there are labels for the target domain, but only used for testing.

2.2 Related Work

DA has been studied in various fields, such as natural language processing for sentiment analysis [4], health care for disease diagnosis [15], and computer vision [7] for object detection [2] and semantic segmentation [27]. Also, there are many types of DA to conquer different scenarios. For instance, semi-supervised domain adaptation where a small amount of labeled target domain data is provided is a common setting [18, 23]. In this paper, we focus on UDA [9, 16] and make a comparison between two common settings.

Most UDA researchers put emphasis on covariate shift setting, which assumes that $P_S(X)$ is different from $P_T(X)$. Among these methods, we can roughly divide them into two main approaches. One is the re-weighting method. The goal of this kind of method is to estimate the importance weight $P_T(X)/P_S(X)$ for each source data. After obtaining the importance weights, they can further do importance-weighted empirical risk minimization to adapt their model to the target domain. Different methods estimate the importance weight differently. [20] utilizes the Kullback-Leibler divergence and some [8, 25] borrow the concept of kernel mean matching [6] to estimate the weight. The other method trying to deal with covariate shift is the adversarial training method [3, 12, 13, 19]. Inspired by the Generative Adversarial Network (GAN) [5], the adversarial training method tries to learn a disentangle embedding by making use of discriminator. With these disentangle embedding features that are domain invariant, they can reduce the distribution difference between the source and target domains under covariate shift setting.

Another setting named label shift is assumed that $P_S(Y) \neq P_T(Y)$. In this setting, previous works mostly utilize the re-weighting method to solve the problem. But different from covariate shift, they try to estimate the importance weight $P_T(Y)/P_S(Y)$. The concept of kernel mean matching can spread to label shift setting [26]. However, time-consuming is the drawback of the re-weighting based method, because it requires calculating the inversion of the kernel matrix which would be dependent on data size. Therefore, it is hard to extend to large scale scenarios. Recently, [1, 11] proposes the method by exploiting an arbitrary classifier to estimate the importance weights and thus can easily be applied to large scale scenarios.

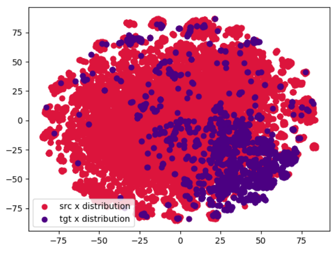
Motivated by a real-world application, we find that current methods cannot successfully tackle this application which contains the properties from covariate and label shift. Therefore, how to promote domain adaptation methods to handle more general cases is essential. Recently, [24] raises a problem that the adversarial training method would cause a bad generalization to the target domain when there exists label shift simultaneously, and proposed the method to handle this.

3 Motivation

Commissioned by the Industrial Technology Research Institute (ITRI), we initiated a research project on predicting compound-protein interaction (CPI), which is a vital topic on drug discovery [10]. Briefly speaking, given a pair of compound and protein data, the CPI prediction task identifies whether the pair comes with chemical interaction or not. That is, the task is a classic binary classification problem. Our collaborators at ITRI provides us with the CheMBL dataset that contains 645461 pairs of (compound, protein), with a binary label for each pair. Note that each example was generated according to the earlier work [22] to obtain a 300-dimensional feature. Each feature is formed by concatenating a 200-dimension compound feature and a 100-dimension protein feature. Additionally, they also indicated 3916 data pairs that are relative to Chinese medicine, named Herb. They hope to get a model having good accuracy on Chinese medicine data. The main difficulty they confront is that labeled Herb data is relatively less compared with CheMBL data. However, doing the experiments to label the data is time-consuming and burning up a lot of money. How to take advantage of a bunch of labeled CheMBL- (CheMBL - Herb) data becomes important in this situation.

We plot the scatter diagram through t-SNE [14] to analyze the dataset. From Figure 1, we can find the distribution of CheMBL- is different from the one of Herb. This figure demonstrates a typical dataset shift scenario. Therefore, we formulate the whole problem as UDA to meet the situation where gathering labeled target data is difficult. As stated above, we have to assume that the source domain and target domain share some properties. Thus, we consider two main assumptions below.

Fig. 1. Visualization for CheMBL-(red) and Herb (purple) by t-SNE



3.1 Covariate Shift Assumption

In this setting, it assumes the input distributions change between source and target domain ($P_S(X) \neq P_T(X)$) while the conditional label distributions remain invariant ($P_S(Y|X) = P_T(Y|X)$). Figure 1 shows that our dataset meets these assumptions so we do the experiments under this setting first. Early works try to estimate the difference between $P_S(X)$ and $P_T(X)$. We call this kind of method re-weighting. Recently, domain adaptation researchers use adversarial training,

Table 1. $P_T(Y)/P_S(Y)$ importance weights estimation between the CheMBL- and Herb.

	class 0	class 1
ground truth	2.3685	0.3130
RLLS	0.0000014	1.0348

utilizing the concept of GAN, to learn a shared transform function E which maps source and target domain data into the same embedding space reducing the distribution difference. We simply do the experiment utilizing Domain Adversarial Neural Network (DANN) [3], a classical adversarial training method, There exists three main components inside the architecture: (i) encoder, (ii) classifier, (iii) discriminator. The encoder E is responsible for mapping the original data to the embedding space Z, where $E : X \to Z$ and try to fool the discriminator so that it can not distinguish between the source and target embedding. The goal of the classifier is to predict well on the source embedding data $C : Z \to \{0, 1\}^K$. What Discriminator do is to verify correctly on the source and target embedding generating from Encoder $E : Z \to \{0, 1\}$. The overall optimization is

$$\min_{E,C} \max_{D} L_{cls}(C, E, \mathbb{D}_S) + L_{adv}(D, E, \mathbb{D}_S, \mathbb{D}_T)$$

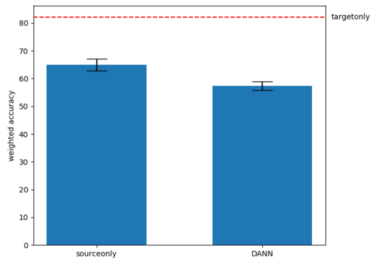
$$= \frac{1}{n}\sum_{i=1}^{n}[y_i^T \log\ C(E(x_i))] + \frac{1}{n}\sum_{i=1}^{n}\log[D(E(x_i))] + \frac{1}{m}\sum_{j=1}^{m}\log[1 - D(E(\tilde{x}_j))]$$

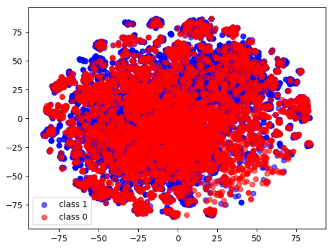
where L_{cls} represents a cross-entropy loss for the source data and L_{adv} is the objective function for encoder and discriminator.

We also train a model on the source domain and directly test it on the target domain, which is called source-only. target-only means that we train the model on training target data then evaluate it on testing target data. Note that, we choose weighted accuracy as the evaluation criterion on Herb dataset because it is an imbalanced dataset.

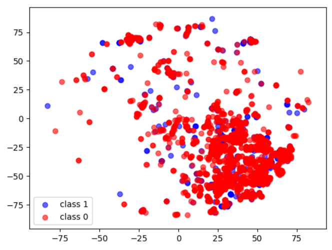
In Figure 2, we notice that of DANN is worse than source-only. Confounding by the result, we dig deeper to analyze the property of this dataset. One possible reason is if we let $P_S(E(X)) = P_T(E(X))$, we can derive $P_S(Y) = P_T(Y)$ based on covariate shift assumption. However, we find that the positive to negative ratio of the number of data is 2:1 in the source domain. In the target domain, the corresponding ratio is 1:4. This finding shows that the label distribution of the source domain is different from the one of the target domain, i.e., $P_S(Y) \neq P_T(Y)$. In this circumstance, if we insist on aligning source and target distribution, we may have bad accuracy. Based on this result, we argue that current adversarial methods designed under covariate shift assumption cannot handle the situation where $P_S(Y)$ is also not equal to $P_T(Y)$.

Fig. 2. Weighted accuracy on Herb





(a) CheMBL-



(b) Herb

Fig. 3. Visualization of label distribution for CheMBL- and Herb

3.2 Label Shift Assumption

It makes the following assumptions. First, the label distribution changes from source to target (i.e. $P_S(Y) \neq P_T(Y)$). Then it further assumes that the conditional feature distributions stay the same ($P_S(X|Y) = P_T(X|Y)$). Recent works deal with this problem through re-weighting and do importance-weighted empirical risk minimization after getting the weights.

$$\begin{aligned}\mathbb{E}_{x,y\sim P_T(X,Y)}\ \boldsymbol{\ell}(y,h(x)) &= \mathbb{E}_{x,y\sim P_S(X,Y)}\ \frac{P_T(X,Y)}{P_S(X,Y)}\boldsymbol{\ell}(y,h(x)) \\ &= \mathbb{E}_{x,y\sim P_S(X,Y)}\ \frac{P_T(Y)}{P_S(Y)}\boldsymbol{\ell}(y,h(x)) \\ &= \mathbb{E}_{x,y\sim P_S(X,Y)}\ w(y)\boldsymbol{\ell}(y,h(x)).\end{aligned} \tag{1}$$

where$h$ stands for a classifier: $x \rightarrow \{0,1\}^K$, $\boldsymbol{\ell}$ represents the loss function: $y \times y \rightarrow [0,1]$ and $w(y)$ denotes the importance weight vector which stands for $P_T(Y)/P_S(Y)$.

We take Regularized Learning under label Shifts (RLLS) [1] as our baseline. The results are reported in Table 1. The table shows RLLS couldn't estimate well on the importance weight. To analyze what takes place in the experiment and cause this bad estimation, we plot figures displaying the source and target distribution separately to observe. According to Figure 1, we are able to confirm that the conditional input distributions are quite different, i.e., $P_S(X|Y) \neq P_T(X|Y)$. This observation breaks the label shift assumption.

3.3 C2H Dataset

From the previous discussion, we found that current domain adaptation methods could not cover all various dataset shift cases, e.g., our real-world dataset. We first formally construct the C2H dataset for this particular domain adaptation task. It comprises CheEMBL- and Herb represented as source and target domain respectively. The source domain includes 641,545 data, and the target domain contains 3,916 data. Specifically, both input and label distributions vary between source and target.

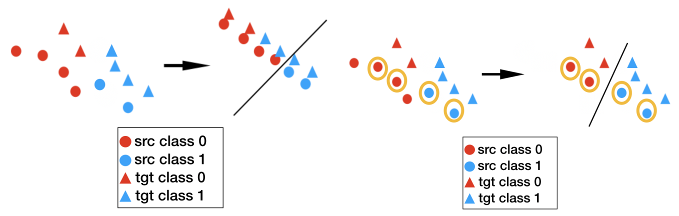
4 Proposed Method

Based on the observations in section 3.1, if we stop at nothing to use adversarial training to align the source and target data distributions, we may finally get an unexpected bad performance on the target domain due to neglecting that there could also exist label shift at the same time. We illustrate the toy example in Figure 4 to demonstrate this finding. In Figure 4(a), if we use adversarial training to align two distributions and do not take $P_S(Y) \neq P_T(Y)$ into consideration, we would probably have bad accuracy on target data. Figure 4(b) shows that when the embedding space has strong physical meaning, selecting the source data which is close to target data directly could get some benefit on classification. That is, we can regard the distance between two data as an similarity measurement and then accomplish domain adaptation through selection technique. We use a toy example to demonstrate our idea in the following section.

4.1 Representatives Selection

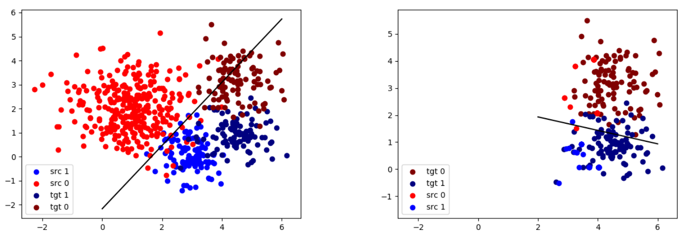
In this section, we demonstrate that a simple selection technique could accomplish UDA in Figure 4. Figure 5(a) depicts the source-only classifier. We can see that directly applying the source-only classifier to the target domain could have a bad performance. In Figure 5(b), choosing the source data which is close to target data and utilizing it could get a good classifier on the target domain, i.e., achieve domain adaptation. Therefore, when the distance can represents similarity, simple selection technique can improve the performance in UDA task.

Furthermore, we actually implicitly make the continuity assumption, i.e., the points which are close to each other are more likely to share the same label. If the assumption holds, we can achieve domain adaptation through selecting the target-like source data which is close to target data. We define the target-like source data as representatives in this paper. Based on the continuity assumption, we further propose two selection techniques to achieve domain adaptation.



(a) adversarial training (b) data selection

Fig. 4. The intuition and illustration of our proposed method



(a) source-only (b) data selection

Fig. 5. Toy example to demonstrate domain adaptation can be done through selection technique

4.2 Hard Distance-Based Selection (HS)

The first method is based on K-Nearest Neighbor (KNN), a classic lazy algorithm. KNN takes euclidean distance as a similarity measurement and collaborates with the assumption that for any data point and its neighborhood must belong to the same class, i.e., continuity assumption. We feed the source data into KNN as training data first and then input all the target data to get the corresponding representatives. We let K = 1 for simplicity. The procedure can be formulated from a different perspective as

$$\text{for each } \tilde{x}_j, \text{ let } s_j = \underset{x_i \in \mathcal{D}_S}{\arg\min} ||\tilde{x}_j - x_i||_2^2,$$

$$\mathcal{D}_S^{rep} = \{s_j\}_{j=1}^m,$$

where $\mathcal{D}_S^{rep}$ denote the representatives we choose. After gathering the representatives, we can use them as a new source dataset to train a model and apply it to the target domain.

4.3 Soft Distance-Based Selection (SS-β)

However, HS could aggravate bad performance when there exist two problems. First, the continuity assumption could be wrong. For example, in high dimensional space like image space, directly take distance as a similarity measurement to select the representatives may be a catastrophe. In this kind of space, the data sparsity problem exists naturally. We may face that the distance does not represent a sort of similarity. Second, if the source data is noisy, choose the representatives by distance may bring a lot of biases into the model and thus hurt the performance. Thus, to overcome these two problems, we propose the second method called SS-β. The soft means we do not drop the rest of the source data after selecting the representatives. Instead, we add the following constraint into the minimization objective. Supposed we train a neural net N as a classifier with L layers, we add the following constraint on the k-th hidden layer

$$\min_{f} \frac{1}{m} \sum_{j=1}^{m} ||N^k(s_j) - N^k(\mathcal{D}_{S_j}^{rep})||_2^2.$$

Via this term, we enforce that the close data pair in original space must be close in embedding space. The overall objective can be

$$\min_{N} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{\ell}(N(x_i), y_i) \ + \ \beta \frac{1}{m} \sum_{j=1}^{m} ||N^k(\tilde{x}_j) - N^k(s_j)||_2^2,$$

where β is a hyperparameter to control the importance of this constraint and $\boldsymbol{\ell}$ represents a cross entropy loss.

5 Experiments

In this section, we evaluate our proposed methods on three parts: (i) C2H, (ii) Noisy C2H and (iii) Digits. For part (i), we want to show simple selection based methods can improve the performance in our C2H dataset. In part (ii), we test our methods in the noisy source domain and discuss what is the best circumstances for our methods to be used. To evaluate the scalability of our methods to high-dimension space, we do the experiments on digits dataset and show the results in part (iii)

We name our methods as follows: (1) HS: use the representatives selected by Hard Distance-Based Selection to train the model and then direct apply it to the target domain. (2) SS-β: train the model on the source domain and add the Soft Distance-Based Selection constraint which is controlled by the hyperparameter β to restrict the influence of this term. For each result, we repeat 5 times trials with different random seeds and show the average on the table. We also indicate the standard deviation to demonstrate the stability for each method.

5.1 C2H Dataset Evaluation

We run the following methods as our competitors (i) KMM-γ [8], the classic re-weighting method and the γ represents the parameter in the Gaussian kernel, (ii) DANN [3], (iii) fDANN-β and sDANN-β proposed by [24] which implicitly deals with the same problem as we do. β is a restrictive factor that forces the model not to perfectly align source and target data. source-only and target-only are also placed as the baselines. Note that, we subsample 20000 data points from ChEMBL- for efficient evaluation. We all use Adam as the optimizer with 512 and 64 batch size for the source and target data respectively, set the learning rate=0.0001. For DANN-like models, it is noteworthy that encoder, discriminator, and classifier have their optimizer with different weight decay (0.01, 0.001, 0, respectively). Figure 6 and Figure 7 shows the architecture of our methods and DANN-like methods respectively.

Fig. 6. Model architecture of HS and SS-β in C2H task

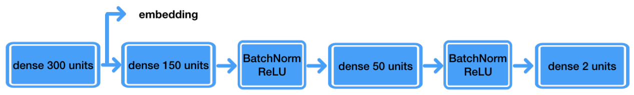


Fig. 7. Model architecture of DANN-like methods in C2H task

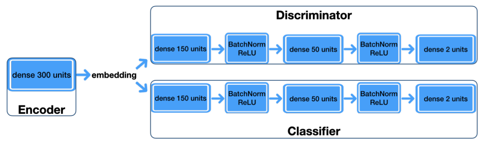
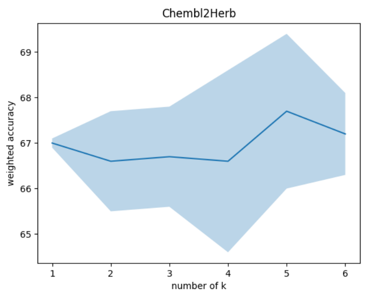


Table 2 shows that HS has an improvement compared with source-only and other methods in this task. We can see that there is a big performance gap between DANN-like methods and ours. The original dataset already has interpretable and discriminative features. Therefore, aligning the distributions aggressively would lead to declining performance, not to mention label shift would deteriorate the performance too. fDANN and sDANN are expected to somewhat ease the impact of label shift by restricting the model not to align the distribution perfectly, but still have bad performance due to destroying the good feature embedding. In Table 2, we can see that re-weighting methods are competitive to HS. HS can basically be regarded as a re-weighting method that only assign the weight to the representatives and others assign 0 weight. However, HS is computational efficiency because we don't need to calculate the kernel matrix that KMM should do. We just run the KNN algorithm. We can also see that our SS-β perform poorly because it suffers from difficult hyperparameter tuning.

Furthermore, we do the experiment to see whether accuracy under the different number of neighbors k would change. The results plot in Figure 8. We can find that under different k the accuracy has slightly different. Therefore, we do not have to worry about the parameter k when using our methods.

Fig. 8. different number of neighbors k



	accuracy
source-only	65.0 ± 2.1
KMM-1	66.7 ± 1.5
KMM-10	66.2 ± 1.0
KMM-100	67.0 ± 1.1
DANN	57.4 ± 1.6
fDANN-1	56.3 ± 1.6
sDANN-4	57.8 ± 1.7
HS	**67.0 ± 0.1**
SS-10	62.3 ± 1.5
target-only	82.2 ± 1.1

Table 2. Weighted accuracy on C2H task.

5.2 Noisy C2H Dataset Evaluation

We want to verify that SS-β could handle the situation where the representatives could be disruptive due to the noisy source data. Therefore, we create a noisy C2H dataset and try to choose the better method in this scenario. First, we add Gaussian noise with 0 mean and 0.01 variance into each feature dimension independently for every ChEMBL- data point, while Herb dataset remains the same.

The experiment results are listed in Table 3. From the table, we can see that HS perform poorly than SS-β. As expected, in the noisy source scenario, if we over-rely on the close source dataset selected by HS, we would suffer from the impact of noisy data. In this circumstance, choose SS-β can mitigate the noisy data effect by careful hyperparameter tuning.

Briefly, if we know in advance that the data has strong physical meaning in your task, use the hard version would gain much more benefit without the effort for tuning the parameter. On the contrary, in the task where source data could have some noises, choose soft version selection and coupe with careful parameter search can avoid over-confidence on the fake representative.

	[0-9] No-Shift
source-only	56.9 ± 1.2
HS	55.6 ± 1.1
SS-1	57.7 ± 1.3
target-only	82.2 ± 1.1

Table 3. Weighted accuracy on noisy C2H task.

5.3 Digit Dataset Evaluation

To extend to a more severe shift scenario, we follow the procedure of previous work [24] to artificially generate the shift datasets. In brief, the source domain keeps class-balanced and the shift part comes from the target domain. To yield the target label distribution shift, we subsample target data from half of the classes in a uniform sampling manner. Therefore, following the procedure, we obtain a covariate shift dataset with severe label shift. We consider USPS and MNIST datasets, so there would be two tasks: (i) USPS $\rightarrow$ MNIST and (ii) MNIST $\rightarrow$ USPS. For each task, we do the following experiments. (a) [0-4] shift: target data only sample from class 0-4. (b) [5-9] shift: target data only sample from class 5-9. (c) [0-9] no shift: sample data from all classes. Note that, we subsample 2000 data from MNIST and subsample 1800 data from USPS according to given distribution (shift or no shift), resize all the image to 28x28, convert each value into [0, 1] and do channel-wise normalization with 0.5 mean and 0.5 standard deviation. Figure 9 and Figure 10 depict the model architectures.

Fig. 9. Model architecture of HS and SS-β in Digit task

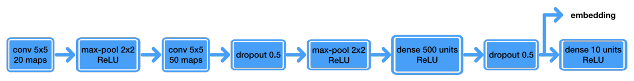
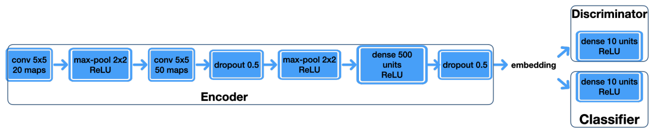


Fig. 10. Model architecture of DANN-like methods in Digit task



For task (i), the results are listed in Table 4. From the table, we can discover that fDANN outperforms other methods on the severe shift setting (i.e. [0-4] Shift and [5-9] Shift). As we expected, our distance-based methods perform ordinary or even worsen because the features do not have great physical meanings. But we can also find out that fDANN and sDANN are unstable with high standard deviations. Therefore, it is not certain whether applying fDANN and sDANN for a real-world application is suitable.

For task (ii), Table 5 shows that fDANN still does well in severe shift settings. However, to our surprise, SS-1 has a great improvement on [0-4] Shift. We further investigate this phenomenon by plotting the source and target distributions in Figure 11. We can find that class 0-4 from both MNIST and USPS have great discriminability because they separate obviously. Additionally, the source data with the labels among class 0-4 is relatively close to the corresponding target data. Therefore, our method can have great performance in [0-4] Shift.

Even though our methods perform well only on [0-4] Shift, the performance of our methods on other tasks is still worse than other methods. Therefore, obtaining a feature embedding with physical meaning is crucial before applying our methods. We try three different ways to get an embedding: (1) PCA: concatenate both the source and target data and then run the method to obtain the features, (2) extractor: build a model from the source domain first, then use it as feature extractor on the source and target data, (3) ImageNet: use ImageNet pre-trained model as a feature extractor. After getting all feature embedding, we then apply our methods on these embedding.

Table 6 and Table 7 show the results. We can see that our method well generalizes to the target domain, under the feature embedding generating by the extractor method. Using the features generated by PCA to run our methods has bad performance on each task. This result shows PCA lets the target data lose a lot of important information. The ImageNet method performs poorly, either. Because it was trained on a non-digits dataset, the model can not extract the features which are important for digits classification.

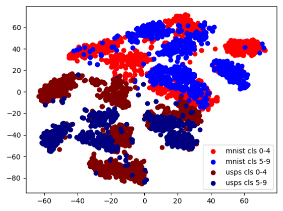


Fig. 11. t-sne of MNIST (src) and USPS (tgt)

	[0-4] Shift	[5-9] Shift	[0-9] No-Shift
source-only	73.1 ± 4.5	29.2 ± 3.3	50.1 ± 3.0
DANN	62.1 ± 1.9	38.8 ± 4.0	**88.6 ± 1.5**
fDANN-1	**74.2 ± 2.2**	**69.5 ± 7.8**	82.1 ± 1.8
sDANN-1	71.7 ± 2.3	42.0 ± 3.5	84.8 ± 1.3
HS	72.3 ± 5.4	26.4 ± 5.4	43.2 ± 4.7
SS-100	71.3 ± 3.2	25.8 ± 3.0	42.9 ± 4.1
SS-10	69.9 ± 2.8	25.9 ± 4.3	41.8 ± 3.8
SS-1	69.7 ± 3.9	26.0 ± 5.2	45.7 ± 4.1
SS-0.1	70.5 ± 2.5	26.9 ± 5.0	48.5 ± 5.2
SS-0.01	73.0 ± 3.1	28.6 ± 3.4	50.2 ± 3.2

Table 4. Accuracy on USPS → MNIST with different label shift settings

	[0-4] Shift	[5-9] Shift	[0-9] No-Shift
source-only	83.5 ± 1.5	58.3 ± 4.4	71.2 ± 2.2
DANN	48.9 ± 4.3	39.2 ± 1.8	**87.0 ± 1.4**
fDANN-1	81.7 ± 2.3	**72.1 ± 7.7**	84.2 ± 3.7
sDANN-4	61.5 ± 8.4	42.4 ± 6.4	82.7 ± 2.5
HS	85.2 ± 0.1	47.5±9.7	70.1±1.4
SS-100	87.3 ± 1.1	58.1 ± 2.3	75.5 ± 0.9
SS-10	88.4 ± 1.3	60.7 ± 2.1	76.3 ± 1.0
SS-1	**88.8 ± 1.2**	62.6 ±1.7	76.7 ± 0.8
SS-0.1	87.7 ± 1.4	62.9 ± 2.2	77.3 ± 0.8
SS-0.01	84.8 ± 1.5	59.7 ± 3.0	74.6 ± 0.9

Table 5. Accuracy on MNIST → USPS with different label shift settings

	[0-4] Shift	[5-9] Shift	[0-9] No-Shift
pca	29.2 ± 2.9	14.7±6.6	23.6±3.7
extractor	83.6 ± 5.3	55.8 ±6.9	69.3 ± 1.7
ImageNet	43.9 ± 3.3	26.9 ±3.2	34.0 ± 3.1

Table 6. Accuracy on MNIST → USPS with different label shift settings under three embedding methods

	[0-4] Shift	[5-9] Shift	[0-9] No-Shift
pca	24.9 ± 2.7	4.7±1.4	13.9±2.5
extractor	77.1 ± 5.3	51.4 ±9.1	67.4 ± 4.2
ImageNet	43.9 ± 3.7	18.8 ±1.7	30.6 ± 2.4

Table 7. Accuracy on USPS → MNIST with different label shift settings under three embedding methods

6 conclusion

Motivated by the real-world bio-chemistry application, we indicate the problem that covariate shift and label shift could exist at the same time. We propose HS and SS-β which can handle this situation while other recent UDA methods would suffer from. We also extend our methods to image space which is high-dimensional. Our methods are mainly based on the similarity, that is, how to get a feature space with strong physical meaning would be a big problem. A possible extension of this work is regarding our methods as a complement for current domain adaptation methods.

References

1. Azizzadenesheli, K., Liu, A., Yang, F., Anandkumar, A.: Regularized learning for domain adaptation under label shifts. In: International Conference on Learning Representations (2019)
2. Chen, Y., Li, W., Sakaridis, C., Dai, D., Gool, L.V.: Domain adaptive faster R-CNN for object detection in the wild. CoRR **abs/1803.03243** (2018)
3. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. Journal of Machine Learning Research **17**(1), 2096–2030 (2016)
4. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: A deep learning approach. In: Proceedings of the 28th international conference on machine learning (ICML-11). pp. 513–520 (2011)
5. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
6. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.: A kernel two-sample test. Journal of Machine Learning Research **13**(Mar), 723–773 (2012)
7. Hoffman, J., Tzeng, E., Park, T., Zhu, J.Y., Isola, P., Saenko, K., Efros, A., Darrell, T.: CyCADA: Cycle-consistent adversarial domain adaptation. In: Proceedings of the 35th International Conference on Machine Learning. vol. 80, pp. 1989–1998 (2018)
8. Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., Smola, A.J.: Correcting sample selection bias by unlabeled data. In: Advances in Neural Information Processing Systems 19, pp. 601–608 (2007)
9. Kang, G., Jiang, L., Yang, Y., Hauptmann, A.G.: Contrastive adaptation network for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
10. Keiser, M.J., Setola, V., Irwin, J.J., Laggner, C., Abbas, A.I., Hufeisen, S.J., Jensen, N.H., Kuijier, M.B., Matos, R.C., Tran, T.B., et al.: Predicting new molecular targets for known drugs. Nature **462**(7270), 175–181 (2009)
11. Lipton, Z., Wang, Y.X., Smola, A.: Detecting and correcting for label shift with black box predictors. In: Proceedings of the 35th International Conference on Machine Learning. vol. 80, pp. 3122–3130 (2018)
12. Long, M., CAO, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: Advances in Neural Information Processing Systems 31, pp. 1640–1650 (2018)
13. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. In: Proceedings of the 34th International Conference on Machine Learning. vol. 70, pp. 2208–2217 (2017)
14. van der Maaten, L., Hinton, G.: Visualizing high-dimensional data using t-sne. Journal of Machine Learning Research **9**, 2579–2605 (2008)
15. Moradi, E., Gaser, C., Huttunen, H., Tohka, J.: Mri based dementia classification using semi-supervised learning and domain adaptation. In: MICCAI 2014 Workshop Proceedings, Challange on Computer-Aided Diagnosis of Dementia, based on Structural MRI Data (2014)

16. Pizzati, F., Charette, R.d., Zaccaria, M., Cerri, P.: Domain bridge for unpaired image-to-image translation and unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (March 2020)
17. Quionero Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D.: Dataset shift in machine learning (2009)
18. Saito, K., Kim, D., Sclaroff, S., Darrell, T., Saenko, K.: Semi-supervised domain adaptation via minimax entropy. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
19. Sankaranarayanan, S., Balaji, Y., Castillo, C.D., Chellappa, R.: Generate to adapt: Aligning domains using generative adversarial networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
20. Sugiyama, M., Suzuki, T., Nakajima, S., Kashima, H., von Bünau, P., Kawanabe, M.: Direct importance estimation for covariate shift adaptation. Annals of the Institute of Statistical Mathematics **60**(4), 699–746 (2008)
21. Wachinger, C., Reuter, M.: Domain adaptation for alzheimer's disease diagnostics. Neuroimage **139**, 470–479 (2016)
22. Wan, F., Zeng, J.M.: Deep learning with feature embedding for compound-protein interaction prediction. bioRxiv (2016). https://doi.org/10.1101/086033
23. Wang, W., Wang, H., Zhang, Z., Zhang, C., Gao, Y.: Semi-supervised domain adaptation via fredholm integral based kernel methods. Pattern Recognition **85**, 185 – 197 (2019). https://doi.org/https://doi.org/10.1016/j.patcog.2018.07.035, http://www.sciencedirect.com/science/article/pii/S0031320318302747
24. Wu, Y., Winston, E., Kaushik, D., Lipton, Z.: Domain adaptation with asymmetrically-relaxed distribution alignment. In: Proceedings of the 36th International Conference on Machine Learning. vol. 97, pp. 6872–6881 (2019)
25. Yu, Y.L., Szepesvári, C.: Analysis of kernel mean matching under covariate shift. In: Proceedings of the 29th International Coference on International Conference on Machine Learning. pp. 1147–1154. ICML'12 (2012)
26. Zhang, K., Schölkopf, B., Muandet, K., Wang, Z.: Domain adaptation under target and conditional shift. In: International Conference on Machine Learning. pp. 819–827 (2013)
27. Zou, Y., Yu, Z., Vijaya Kumar, B., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 289–305 (2018)

On the Transferability of Deep Neural Networks for Recommender System

Duc Nguyen[1], Hao Niu[1], Kei Yonekawa[1], Mori Kurokawa[1], Chihiro Ono[1], Daichi Amagata[2], Takuya Maekawa[2], and Takahiro Hara[2]

[1] KDDI Research Inc., Japan
{du-nguyen,ha-niu, ke-yonekawa, mo-kurokawa, ono}@kddi-research.jp
[2] Osaka University, Japan
{amagata.daichi, maekawa, hara}@ist.osaka-u.ac.jp

Abstract. Recommender system is an essential component in many practical applications and services. Recently, significant progress has been made to improve performance of recommender system utilizing deep learning. However, current recommender systems suffers from the long-standing data sparsity problem, especially in domains with little data. With the ability to transfer knowledge across domains, transfer learning is a potential approach to deal with the data sparsity problem in recommender system. In this paper, we carry out an investigation on the transferability of deep neural networks for recommender system. We show that network-based transfer learning can improve recommendation performance on target domains by up to 20%. In addition, our investigation reveals that transferring the layers close to the output leads to better transfer performance. The transfer performance is also found to be dependent on the similarities between data distributions of the source and target domains. Meanwhile, target domain characteristics such as size and sparsity have little impacts on the transfer performance.

Keywords: Transfer Learning, Recommender System, Neural networks

1 Introduction

With the explosive growth of information available on the Internet, it is challenging for users to find their desired products/services. Thus, recommender systems (RSs) play a central role in enhancing user experience, especially in online news services, E-commerce websites, and online advertising [24]. The main task of RSs is to provide suggestions for items (e.g., news, books, movies, event tickets, etc.) to individual users. RSs enable the so-called *personalized experience*, which is the key to the successes of many Internet companies like Amazon [28], Netflix [8].

Starting with the Netflix Prize [3], significant progress has been made in recommender system research [33]. The past few years have also witnessed the great success of deep learning in many application domains, especially in computer vision and natural language processing [15]. In this trend, in the past few years, deep learning has been studied extensively for recommender system such as in [1, 4, 10, 13, 25, 26, 34]. Although these deep learning-based methods are effective in improving the performance of recommender system, they are mostly based on information (e.g., ratings, reviews) in a single domain. As a result, these methods inevitably suffer from the data sparsity problem because each item is usually rated or reviewed by a few users [24]. Moreover, current applications should be able to react quickly to new situations such as new products or new users. Therefore, techniques to reuse knowledge across times, domains, and tasks are highly desirable.

Transfer learning is a machine learning technique capable of transferring knowledge learned in a domain (*source domain*) to another related domain (*target domain*) [22]. Thus, it can be used to deal with the data sparsity problem in recommender system as well as to increase system's ability to adapt to new situations. Existing works on transfer learning for recommender system apply either *instance-based* [5, 7, 16, 23] or *feature-based* [19, 35] approaches, in which data samples/features from one or more source domains are transferred to a target domain. One of the main problems of instance-based and feature-based transfers is that they require access to data of other source domains. In other words, data sharing between domains is necessary. Nevertheless, inter-domain data sharing has become more and more difficult nowadays due to data regulations such as GDPR [29], especially if the shared data contains user-relevant information.

To improve the performance of recommender systems, it is still desirable to be able to transfer knowledge across domains even if shared data is not available. In such circumstances, *network-based transfer learning*, which transfers features of model (e.g., parameters, structure, etc.) learned on a source domain, is a potential approach. Although network-based transfer has been studied extensively in the literature, previous works mainly focus either on computer vision [9,14,21,27,30,32] or natural language processing [6,11,12,18,31]. In context of recommender systems, despite the fact that deep neural network-based models have shown their superiority, there is still no existing work on the transferability of those deep neural networks.

In this paper, we focus on answering the following three questions in order to understand the transferability of neural network for recommender system.

– Q1: Does network-based transfer learning lead to better recommendation performance on the target domain?
– Q2: How to transfer a neural network for the best transfer performance?
– Q3: What are the factors affecting the transfer performance?

Although network-based transfer learning has been found to be effective in many computer vision and natural language processing(NLP) tasks, there is still a lack of understanding on the transferability of neural networks for recommender tasks (i.e., Q1). In computer vision and NLP tasks, those layers close to the input are found to be highly transferable, whereas those close to the output are task-specific [21]. Yet, it is still unknown which layers can be effectively transferred in recommendation tasks (i.e., Q2). It is also important to understand how different factors affect the transfer performance (i.e., Q3).

In this paper, we investigate the transferability of deep neural networks for recommender system, focusing on *top-N item recommendation task*. For that purpose, a recommender system built on Multi-layer Perceptron (MLP) neural network is used as the base network. The base network consists of an embedding layer and an interaction function consisting of multiple fully connected layers. Then, we examine various options to transfer the knowledge of the base network to a target domain. Extensive evaluation with eighteen real-world datasets demonstrate that transferring the interaction function layers can improve recommendation performance on the target domain by up to 20%. Especially, our evaluation reveals that, unlike deep neural networks for computer visions and NLP tasks, those layers close to the output are more transferable than those close to the input in deep neural networks for recommender system. To the best of our knowledge, this is the first work on transferablity of deep neural networks for recommender system.

The remaining of the paper is organized as follows. Section 2 surveys related works. The base network and transfer options are described in Section 3. The evaluation is given in Section 4. Finally, the paper is concluded in Section 5.

2 Related Work

In recent years, deep learning-based methods have been studied extensively for recommender systems. These methods mainly focus on replacing one or more components in conventional methods by deep neural networks. For instance, in [10], instead of using the dot product as in traditional matrix factorization [2], the interaction function is learned by a MLP network. In [25,36], Autoencoder is utilized to learn the user/item embeddings. In [1], Gate Recurrent Unit is used to exploit the order of words in sentences, which is shown to outperform a simple average of word embeddings for text recommendation. Other deep neural network architectures such as Generative Adversarial Network (GAN) [4] and Attention Model [26] have also been used in recommender system. A comprehensive survey of deep learning-based methods can be found in [33]. In this paper, we adopt the MLP as the base network due to its simplicity. Other deep neural networks will be studied in our future work.

In the literature, transfer learning has been used to tackle the data sparsity problem in recommender system. Most transfer learning methods in previous studies are either *instance-based* or *feature-based*. In [5], training samples of a source domain are directly used to train the recommendation model at the target domain. In [16], users/items in a source domain are clustered to construct a codebook, which is then transferred to a target domain. In [19,35], the user/item feature vectors learned on a source domain are transferred to the target domain by means of a mapping function. Some other studies leverage *multi-task learning* to enable dual knowledge transfer across domains such as [13, 34]. However, instance-/feature-based transfers and multi-task learning require sharing data between domains. In contrast, our work focuses on *network-based transfer*, and thus does not require data sharing across domains. Such a property is especially

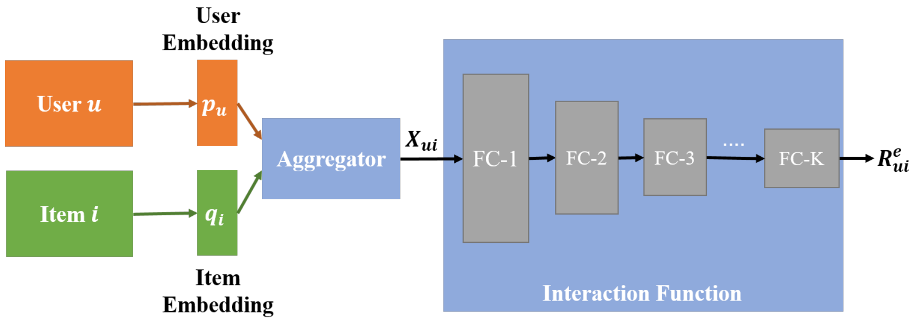


Fig. 1: Base network architecture for top-N item recommendation task.

important considering fact that more data regulations are being imposed on user data [29].

Network-based transfer learning has been studied in contexts of computer vision and natural language processing research. In [32], it is found that the first layers of Convolutional Neural Network (CNN) are highly transferable. The following works lead to the developments of various transfer techniques for image classification task. In [21], the output layer of a pre-trained CNN is replaced by an adaptation layer, while the remaining layers are transferred to the target domain. In [9], only the convolutional layers are transferred, while all the fully-connected layers of CNN networks are fine-tuned with learning rate determined by Bayesian Optimization. A recent evaluation [14] found that there is a strong correlation between ImageNet accuracy and transfer accuracy among popular image classification networks. To improve the performance of factoid question answering (QA) tasks on small datasets, the model parameters trained on a large dataset are used to initialize the target model's weights, with a modified loss function to avoid catastrophic forgetting [31]. In [12], an universal language modeling fine tuning (ULMFiT) is presented, featuring discriminative fine-tuning, slanted trianglar learning rates, and gradual unfreezing. In [11], an adapter-based parameter efficient transfer learning for NLP is proposed.

3 Network-based Transfer Learning

In this section, the top-N item recommendation task is defined and a neural network-based approach is introduced. Then, we describe how to transfer the a pre-trained network from a source domain to a target domain.

3.1 Top-N item recommendation task

Along with rating prediction [3], top-N item recommendation is one of most important tasks in recommender systems. Suppose that we need to recommend N items to individual users of a particular domain (e.g., online book stores, e-commercial websites). Let $\mathcal{U}$ and $\mathcal{I}$ respectively denotes the sets of users and items. We define the variables $\{R_{ui}\}$ to represent user-item interactions as follows.

$$R_{ui} = \begin{cases} 1 & \text{if user } u \text{ has interacted with item } i \\ 0 & \text{otherwise} \end{cases}$$

Here, an interaction can be a purchase or rating of the item, a click on the item's advertisement, or a visit to the item's website. In this paper, we assume that only implicit feedback is available. Thus, user-item interactions are represented by binary values. The set of items that a user u has interacted with in the past is denoted by $\mathcal{I}_u$, i.e., $\mathcal{I}_u = \{i|R_{ui} = 1\}$. The top-N item recommendation problem can be formulated as follows.

For a user $u \in \mathcal{U}$, determine N items $\{i_1, i_2, .., i_N\} \in \mathcal{I} \setminus \mathcal{I}_u$ that have the highest likelihoods that the user u will interact with.

Existing methods for top-N item recommendation task can be classified into two main groups, namely *content-based*, and *collaborative filtering*. Content-based methods simply calculate the similarity between candidate items and the items the user has interacted with, then select top-N items with highest similarity scores. On the other hand, collaborative filtering predicts the interaction score by using preference from many users. In this paper, we focus on model-based CF to predict the value R_{ui} for every item $i \in \mathcal{I} \setminus \mathcal{I}_u$. The model is built on top of a neural network and will be described in the next section.

3.2 Neural Network Model

In this paper, we follow the NeuMF framework proposed in [10] to build the base network as follows. Each user/item is characterized by a latent vector or embedding. The user-item interactions are modeled by an interaction function. Similar to [10], the interaction function is a Multi-layer Perceptron network, which is learned during training.

Figure 1 shows the architecture of the base network used in this paper. As aforementioned, each user $u \in \mathcal{U}$ is characterized by an embedding vector $p_u \in \mathbb{R}^{d_u}$, where d_u is the user embedding size. Similarly, each item $i \in \mathcal{I}$ is mapped to an item embedding vector $q_i \in \mathbb{R}^{d_i}$ where d_i is the length of the item embedding vectors. In this paper, we assume that the user and item embeddings have the same size, i.e., $d_u = d_i$. Given an interaction between user u and item i, the corresponding user and item embeddings are aggregated by the *aggregator*, forming X_{ui}, which is the input of the *interaction function*. In this paper, the aggregator simply concatenates the user and item embedding vectors as follows.

$$X_{ui} = [p_u, q_i] \tag{1}$$

The interaction function consists of K fully connected layers FC-k ($1 \leq k \leq K$). Let s_k denote the size of layer FC-k. The output $y_k \in \mathbb{R}^{s_k}$ of layer FC-k ($1 \leq k \leq K$) is given by,

$$y_k = \begin{cases} f_k(X_{ui} * W_k + b_k) & \text{if } k = 1 \\ f_k(y_{k-1} * W_k + b_k) & \text{if } k > 1 \end{cases} \tag{2}$$

where f_k, $W_k \in \mathbb{R}^{s_{k-1} \times s_k}$, and $b_k \in \mathbb{R}^{s_k}$ respectively denotes the activation function, weight, and bias of layer FC-k. The outermost FC layer (i.e., FC-K) is also referred to as *output layer*. The base network parameter set θ includes the user and item embeddings and the layers' weights and biases.

$$\theta = \{\{p_u\}_{u \in \mathcal{U}}, \{q_i\}_{i \in \mathcal{I}}, \{W_k, b_k\}_{1 \leq k \leq K}\} \tag{3}$$

The parameter set θ is learned so as to minimize a loss L, which is a function of the predicted interaction and the actual ones.

$$\min_{\theta} \frac{1}{|\mathcal{U}| \times |\mathcal{I}|} \sum_{(u,i)} L(R_{ui}, R^e_{ui}) \tag{4}$$

Where R^e_{ui} is the predicted interaction interaction of user u and item i. In this paper, since the interaction values are binary, we adopt the binary cross-entropy loss function.

3.3 Network-based Transfer Learning Mechanism

In this paper, we are interested in the transferability of deep neural networks learned on a source domain to improve performance on a target domain. As aforementioned, since we assume that data sharing is not available, instance-based transfer is not applicable since it requires transferring of data instances from the source domain to the target domain. Feature-based transfer (e.g., [19]) requires prior knowledge of shared users/items, which is unknown in this case, and so cannot be applied. Thus, a network-based transfer approach [23] is used. Given a target domain D_T and a learning task T, the goal here is to improve performance on D_T by transferring knowledge of the pre-trained network learned on a source domain D_S.

The key assumption of network-based transfer approach is that the neural networks of the source and target domains should share some parameters. Let θ_S and θ_T respectively denotes the parameter set of the source and target networks. Then, the parameter sets can be decomposed into two sub-sets, one contains shared parameters (i.e., θ_0) and another contains domain-specific parameters (i.e., v_S and v_T) as follows.

$$\theta_S = \theta_0 \cup v_S \tag{5}$$

$$\theta_T = \theta_0 \cup v_T \tag{6}$$

The common parameters θ_0 are learned on the source domain and then transferred to the target domain. During training at the target domain, the common (transferred) parameters are frozen, whereas domain-specific parameters (v_T) are learned.

Since user/item linkages are not allowed in our problem setting, the user and item embedding vectors are non-transferable, and so they are in domain-specific parameter set v_T. Transferable parameters consists of the weights and

Table 1: Transfer settings of fully-connected layers of the base network.

Setting	**Layers to transfer**
Config-1	FC-1, FC-2, FC-3, FC-4
Config-2	FC-1, FC-2, FC-3
Config-3	FC-1, FC-2
Config-4	FC-1
Config-5	FC-2, FC-3, FC-4
Config-6	FC-3, FC-4
Config-7	FC-4

biases of individual fully-connected (FC) layers of the interaction function. In this paper, we perform transfer in layer basis, in which all parameters of a given layer are transferred as a whole. More fine-grain transfer options are reserved for our future work. We consider different transfer configurations as will described in the next section.

4 Evaluation

4.1 Experiment Setup

Base Neural Network Parameters The user and item embedding sizes are both set to 32. The interaction function consists of $K = 4$ fully connected layers with the sizes of 64, 32, 16, and 8. It should be noted that the size of the first hidden layer of the interaction function network is equal to the sum of the user and item embedding sizes. We compare performance in terms of Hit Ratio (HR) with a baseline in which the base network are trained from scratch using only data in the target domain. For both the transfer options and baseline, Adam optimizer is used. The learning rate is set to 0.001. The batch size is 256. The number of epoch is 100. For each method/option, we run the experiment ten times and report the average values.

Transfer Configurations To investigate the transfer learning performance, we consider seven transfer configurations of the base neural network as shown in Table 1. The configurations differs based on which fully-connected layers are being transferred. It should be noted that the user/item embeddings are not transferable.

Evaluation Protocol To evaluate the proposed method, we follow the leave-one-out evaluation protocol [10]. Specifically, for a user, a test item is randomly chosen among the items that the user have interacted with. In addition, 99 negative items, which have not been interacted by the user, are randomly selected. The predicted scores for the test and negative items are calculated. Then, the test item is ranked against the negative ones based on the predicted scores. The performance metric of hit ratio (HR) is computed as follows. Let h_u denote the hit position (rank) of the test item of user u against the negative items. HR@N is defined as:

$$\text{HR@N} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \max(0, 1 - \lfloor (h_u/(N+1) \rfloor) \tag{7}$$

Here, $\lfloor . \rfloor$ is the floor function. The HR has the range in [0, 1] where a higher value indicates better performance. In this paper, we use HR@10 as the performance metric.

Datasets In our evaluation, eighteen real-world datasets from Amazon Review database [20] are used. The original datasets are preprocessed by removing users and items with less than 20 interactions. Statistics of all datasets are shown in Table 2. We train the base network from the scratch by randomly initializing weights and evaluate the performance on each dataset (i.e., baseline). As can be seen in Table 2, the three datasets of Book, Movie, and Kindle have the highest recommendation performance. Thus, those datasets are chosen as the source domains. The remaining fifteen datasets are taken as target domains.

Table 2: Statistics and baseline (non-transfer) performance of eighteen datasets used in our experiment. The three datasets of Book, Movie, Kindle are taken as source domains.

ID	Dataset	#users	#items	#ratings	sparsity (%)	baseline (HR@10)
1	Book	46276	148785	2453521	99.96	**0.66**
2	Movie	8396	25839	449685	99.79	**0.64**
3	Kindle	13742	21883	566622	99.81	**0.61**
4	Sport	2826	14016	109229	99.72	0.25
5	Clothing	11975	69009	743040	99.91	0.18
6	CD	5019	12847	193170	99.70	0.49
7	Pet	2153	7591	95753	99.41	0.26
8	DigitalMusic	230	2116	7146	98.53	0.20
9	Home	2002	9962	73672	99.63	0.13
10	Toy	2181	9577	66321	99.68	0.34
11	Videogame	514	1996	16496	98.39	0.31
12	Art	531	3492	17804	99.04	0.27
13	Automotive	2196	13435	87418	99.70	0.15
14	Cellphone	146	1726	3813	98.49	0.21
15	Food	1451	6134	59899	99.33	0.21
16	Instrument	173	1271	7185	96.73	0.17
17	Office	543	2768	19309	98.72	0.29
18	Garden	213	1877	5957	98.51	0.20

Table 3: Performance gain (%) of different transfer configurations compared to the baseline (non-transfer) of individual target domains when Book is the source domain. A positive (negative) value means positive (negative) transfer. The last column shows the best configuration and the corresponding $HR@10$.

Target domain	config-1	config-2	config-3	config-4	config-5	config-6	config-7	best config ($HR@10$)
Sport	-11.40	-32.20	-30.68	-13.15	2.64	**3.17**	1.44	config-6 (0.255)
Clothing	**19.90**	-14.81	-19.21	-9.99	4.59	4.26	0.61	config-1 (0.213)
CD	-1.78	-26.51	-28.66	-11.96	-2.15	**0.14**	-1.03	config-6 (0.489)
Pet	-4.41	-26.28	-24.14	-9.26	**0.02**	-0.65	-0.30	config-5 (0.255)
DigitalMusic	-10.74	-11.92	-11.49	-6.17	**-0.43**	-6.06	-9.53	config-5 (0.200)
Home	1.80	-11.67	-12.68	-4.10	**10.89**	8.59	3.81	config-5 (0.143)
Toy	-8.31	-25.06	-23.68	-9.18	-0.88	-1.47	**0.38**	config-7 (0.342)
Videogame	-0.43	-28.77	-27.38	-11.52	0.97	1.07	**1.58**	config-7 (0.312)
Art	2.80	-20.51	-26.08	-4.78	6.47	**9.05**	8.48	config-6 (0.291)
Automotive	-4.59	-17.01	-19.35	-5.25	1.85	**2.74**	0.24	config-6 (0.158)
Cellphone	-19.85	-10.03	-21.30	-17.75	-12.58	**-7.75**	-9.36	config-6 (0.196)
Food	0.35	-26.52	-22.33	-9.76	**4.51**	0.30	-1.15	config-5 (0.218)
Instrument	-13.61	-7.38	-10.14	**0.66**	-3.24	-4.63	-7.38	config-4 (0.168)
Office	**-0.52**	-13.22	-16.68	-5.52	-2.98	-0.80	-2.47	config-1 (0.268)
Garden	-8.29	-14.23	-16.00	-6.65	-4.51	-7.52	**-0.80**	config-7 (0.201)

Table 4: Performance gain (%) of different transfer configurations compared to the baseline (non-transfer) of individual target domains when Movie is the source domain.

Target domain	config-1	config-2	config-3	config-4	config-5	config-6	config-7	best config ($HR@10$)
Sport	0.58	-17.34	-14.62	-12.76	**3.10**	2.35	0.71	config-5 (0.254)
Clothing	**15.42**	-15.49	-9.38	-12.76	7.90	3.29	-1.35	config-1 (0.205)
CD	-0.32	-16.68	-12.33	-10.37	-1.25	**1.65**	-0.15	config-6 (0.497)
Pet	**1.90**	-12.69	-10.25	-10.65	-0.02	-0.72	-1.80	config-1 (0.260)
DigitalMusic	**-0.86**	-3.46	-3.81	-4.73	-2.92	-8.66	-8.66	config-1 (0.199)
Home	**17.58**	-3.11	-2.68	-1.79	14.24	4.90	-0.04	config-1 (0.151)
Toy	**0.89**	-14.11	-8.66	-9.88	-1.41	-2.18	-0.74	config-1 (0.344)
Videogame	**8.18**	-9.63	-9.19	-10.58	1.07	3.61	2.28	config-1 (0.332)
Art	5.95	-9.08	-8.66	-8.30	4.39	5.24	**6.22**	config-7 (0.284)
Automotive	**8.24**	-14.03	-8.84	-7.96	4.82	2.98	0.57	config-1 (0.166)
Cellphone	**6.72**	-3.55	4.52	-0.06	-9.38	-5.48	-5.49	config-1 (0.227)
Food	**10.70**	-10.62	-12.57	-4.30	7.91	3.10	-0.96	config-1 (0.231)
Instrument	**7.41**	-1.24	1.15	-1.45	2.53	-7.38	-8.77	config-1 (0.180)
Office	**5.15**	-4.77	-6.75	-1.83	-0.17	-1.25	-1.76	config-1 (0.303)
Garden	4.74	-1.81	-4.28	-7.05	-3.43	**4.75**	0.81	config-6 (0.213)

Table 5: Performance gain (%) of different transfer configurations compared to the baseline (non-transfer) of individual target domains when Kindle is the source domain.

Target domain	config-1	config-2	config-3	config-4	config-5	config-6	config-7	best config (HR@10)
Sport	-6.04	-16.91	-22.22	-10.57	**3.89**	2.25	1.35	config-5 (0.256)
Clothing	**6.22**	-18.22	-16.46	-10.86	2.22	2.09	0.50	config-1 (0.189)
CD	-5.47	-15.68	-19.36	-9.99	**1.56**	0.96	0.15	config-5 (0.496)
Pet	-3.88	-14.75	-15.53	-10.37	**2.73**	0.05	-0.96	config-5 (0.262)
DigitalMusic	-11.27	-7.58	-7.80	-7.58	**-1.07**	-8.88	-5.62	config-5 (0.198)
Home	2.30	-3.97	-4.59	-2.72	**15.29**	3.70	1.05	config-5 (0.148)
Toy	-9.69	-13.63	-16.36	-8.00	-1.40	**-0.55**	-0.73	config-6 (0.339)
Videogame	0.56	-7.08	-17.78	-9.38	**6.15**	5.20	1.07	config-5 (0.326)
Art	5.54	-6.89	-15.36	-2.22	**11.49**	7.14	6.08	config-5 (0.298)
Automotive	-0.64	-8.28	-10.85	-4.91	**7.49**	0.42	0.79	config-5 (0.165)
Cellphone	-6.11	-2.01	-8.87	-12.09	-6.45	**1.29**	-4.84	config-6 (0.215)
Food	2.43	-16.48	-13.35	-6.42	**8.67**	3.53	0.40	config-5 (0.227)
Instrument	1.07	-4.11	-6.26	-3.66	**1.53**	-2.20	-3.58	config-5 (0.170)
Office	**3.76**	-4.48	-9.84	-5.21	1.11	-1.57	-0.43	config-1 (0.299)
Garden	-10.70	-12.43	-7.52	-8.38	-6.94	-0.35	**0.35**	config-7 (0.204)

4.2 Evaluation Results

In the first part of our experiment, we aim to answer the first and second questions regarding the transferability of the base network, namely *Q1: Does transfer learning lead to better recommendation on the target domain?* and *Q2: How to transfer a neural network for the best transfer performance?*. Table 3, Table 4, Table 5 show the gains of seven transfer configurations compared to the baseline (non-transfer method) of individual target domains when the source domain is Book, Movie, and Kindle, respectively. A positive (negative) value indicates positive (negative) transfer. The last column of each table shows the configuration with the highest gain and the corresponding HR@10.

It can be seen that, for all three source domains, transferring the neural network can improve the performance of most target domains. Among the fifthteen target domains, fourteen domains are benefited from transferring from at lest one source domain. In particular, the number of target domains with positive transfer are 11, 14, and 13 when the source domain is Book, Movie, and Kindle, respectively. Transferring can improve the Hit Ratio on the target domain by up to 20% from the Book domain, up to 17% from the Movie domain, and up to 15% from the Kindle domain. There are 10 target domains in which positive transfer occurs with all three source domains, namely Automotive, Home, Food, Art, Clothing, CD, Pet, Sport, Video, and Instrument. For the domains when the negative transfer occurs, the Hit Ratio is reduced by 1-8%(Book), 1-4%(Movie), and 1-5%(Kindle) compared to the baseline method. For the DigitalMusic domain, transfer learning always causes performance degradation compared to the baseline for both three source domains.

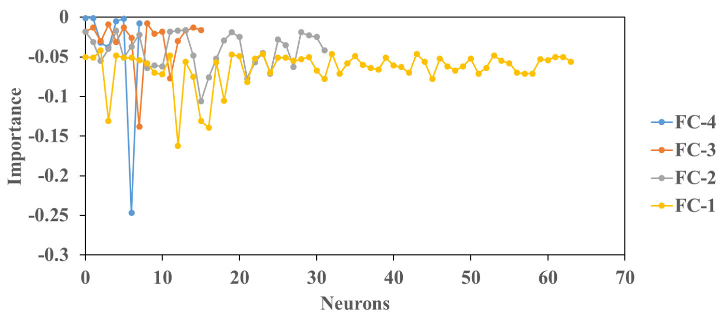


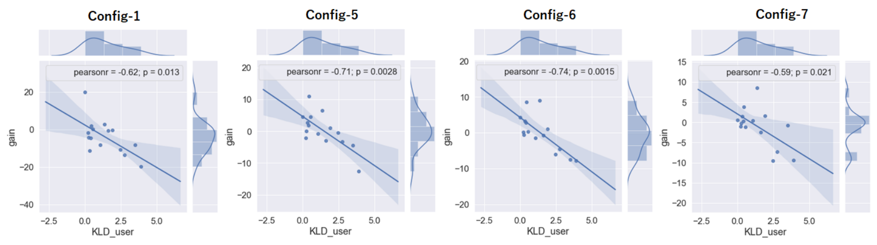
Fig. 2: Importance of individual layers of the source domain model (Book).

It can also be noted that the best transfer configuration varies across target domains and source domains. When Book is the source domain, the best transfer configuration under the Clothing and Instrument domains are Config-1 and Config-4, respectively. For the four domains of Sport, CD, Art, and Automotive, Config-6 yields the highest gains. Especially, Config-2 and Config-3 are in no case the best. Config-4 leads to negative transfer with all target domains except for Instrument. Generally, the performance of those three configurations are 10-30% lower than that of the baseline.

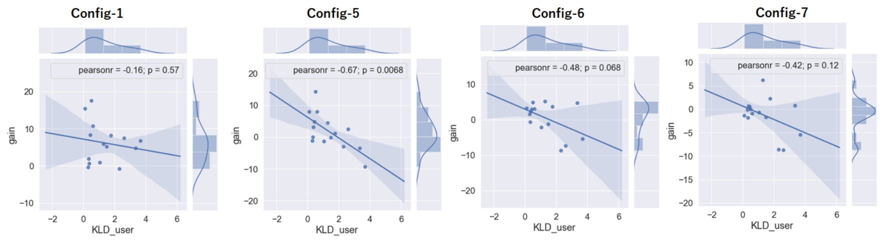
When transferring from the Movie domain (i.e., Table 4), Config-1 is the best configuration for ten target domains. The Config-5, Config-6, and Config-7 configurations are the best configuration for only one target domain domain. Again, it can be seen that the Config-2, Config-3 and Config-4 configurations results in negative transfer for all target domains. As can be seen in Table 5, Config-5 are the best configuration for most target domains when Kindle is the source domain. For the two domains of Clothing and Office, Config-1 achieve the highest gains. Again, it can be seen that the Config-2, Config-3, and Config-4 configurations cause negative transfer in all target domains.

To understand the importance of individual layers, we follow the method proposed in [17] to calculate the importance of individual fully connected (FC) layers. Specifically, to evaluate the importance of a neuron, the log-likelihoods of the correct label with and without the presence of the neuron are compared, and the importance is calculated. Fig. 2 shows the importance values of individual neurons of different FC layers. It can be seen that the layers close to the outputs are generally more importance than those close to the inputs. This result may be a hint to explain why the three configurations of Config-2, Config-3, and Config-4, where the FC-4 is not transferred, are worsen than the other configurations. This issue will be studied further in our future work.

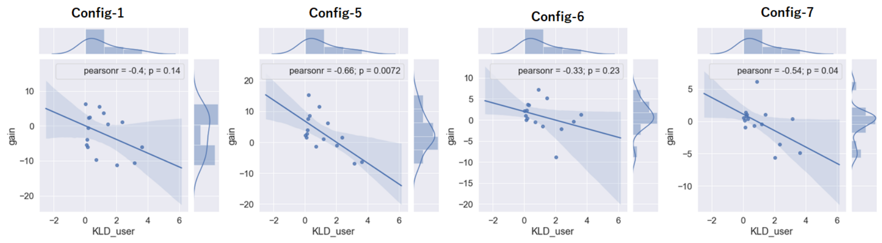
From the above results, we can have the following remarks regarding the transferability of neural networks for recommendation systems.



(a) Source: Book



(b) Source: Movie



(c) Source: Kindle

Fig. 3: Relationship between KLD values and the gains of the Config-1, Config-5, Config-6, and Config-7 configurations for different source domain a) Book, b) Movie, and c) Kindle.

- Transferring the pre-trained network from three source domains of Book, Movie, and Kindle can improve the recommendation performance on most of the target domains.
- For a given source domain, different target domains require different transfer configurations. Especially, Config-1 is preferable when Movie is the source domain, whereas Config-5 achieves highest gains for the highest number of target domains when Kindle is the source domain.
- Config-2, Config-3, and Config-4 always lead to negative transfer. This indicates that transferring of the source model contain the layers close to the output such as in case of Config-1, Config-5, Config-6, and Config-7.

In the second part of our experiment, we investigate how different factors affect the transfer performance, i.e, Question Q3. It is well-known that transfer learning is more effective if the source and target domain are related [32]. Thus, we first examine the impact of the *relatedness* between a source domain and a target domain on the transfer performance. In this paper, we use the similarity between data distributions of the source and target domains to measure the relatedness. For that purpose, we first calculate the histogram of the number of purchases per user H_u of individual domains. Then, we use the KL-Divergence (KLD) to measure the relatedness $R(D^S, D^T)$ between a source domain D^S and a target domain D^T as follows.

$$R(D^S, D^T) = KLD(H_u(D^S), H_u(D^T)) \tag{8}$$

Figure 3 shows the relationship between KLD values and the gains of the Config-1, Config-5, Config-6, and Config-7 configurations for three source domains. The line in each figure show the linear regression fit of the data with an 95% confident interval. The Pearson correlation coefficients (PCC) and p-values are also shown. Because the Config-2, Config-3, Config-4 result in negative transfer for most of the cases, they are excluded in this part. As can be seen in Fig. 3a, when the Book is the source domain, the transfer gain correlates to the KLD values, in which higher KLD value tends to lead to lower transfer learning performance. Especially, this trend is clearly shown in cases of Config-5 and Config-6 where $|PCC| > 0.7$. In case of Movie as the source domain (i.e., Fig. 3b), only the gain of Config-5 shows correlations with the KLD values, whereas the correlations between the three configurations of Config-1, Config-6, and Config-7 are not statistically significant, i.e., *P-value* > 0.05. As for the Kindle domain(i.e., Fig. 3c), the correlation between transfer gain and KLD can be observed for Config-5 and Config-7, but not for Config-1 and Config-6.

Table 6: Pearson correlation coefficients(P-values) between the transfer performance of transfer configurations and a) the target domain sizes and b) target domain sparsity.

(a) Target domain dataset size

Config	Book	Movie	Kindle
Config-1	0.74 (0.002)	0.39 (0.151)	0.35 (0.198)
Config-5	0.28 (0.318)	0.35 (0.205)	0.03 (0.917)
Config-6	0.32 (0.249)	0.26 (0.357)	0.13 (0.640)
Config-7	0.17 (0.533)	0.07 (0.805)	0.14 (0.606)

(b) Target domain sparsity

Config	Book	Movie	Kindle
Config-1	0.48 (0.069)	0.07 (0.801)	0.03 (0.913)
Config-5	0.47 (0.074)	0.35 (0.198)	0.32 (0.252)
Config-6	0.55 (0.032)	0.56 (0.030)	0.31 (0.260)
Config-7	0.54 (0.037)	0.53 (0.040)	0.45 (0.089)

Next, we investigate how characteristics of the target domain affect the transfer performance. Specifically, we consider two key characteristics of the target domain, namely dataset size and sparsity. Table 6 show the correlation coefficients (P-value) between transfer performance and a) target domain dataset size and b) target domain sparsity. It can be seen in Table 6a that the correlation between the transfer performance with the target domain's dataset size is low ($|PCC| < 0.4$) and statistically insignificant ($P\text{-}value > 0.05$), except for Config-1 configuration with Book as the source domain. As shown in Table 6b, the target domain's sparsity has higher correlation to the transfer performance than the dataset size for Config-5, Config-6, and Config-7. However, the PCC values are generally low ($|PCC| < 0.6$). Especially, there is almost no correlation between the transfer performance of Config-1 and the sparsity when Movie and Kindle are source domains.

5 Conclusions

In this paper, we investigate the transferability of deep neural networks for top-N item recommendation task in recommender systems. Specifically, we adopt MLP as the base network, and investigate seven transfer configurations using eighteen real-world datasets. Experimental results shows that transferring layers of the interaction network enhance performance on most of the target domains by up to 20% in terms of Hit Ratio. Especially, in contrast to neural networks for computer vision and NLP tasks, the layers close to the output are more transferable than those close to the input. We also found that the best transfer configuration highly depends on the source and target domains. Hence, different from other tasks such as image classification, the transfer configuration should be carefully chosen to achieve good performance in embedding-based recommendation. Further investigation reveals that the relatedness between the source and target domain measured in terms of KL-Divergence affects the transfer performance, whereas the sizes and sparsity of target domains have little impacts on the transfer performance. In future work, we will focus on developing transfer techniques to dynamically decide the optimal transfer configuration.

Acknowledgment

This research was partially supported by JST CREST Grant Number J181401085, Japan.

References

1. Bansal, T., Belanger, D., McCallum, A.: Ask the gru: Multi-task learning for deep text recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems. pp. 107–114. Boston, Massachusetts, USA (Sep 2016)
2. Bell, R., Koren, Y., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(08), 30–37 (Aug 2009)
3. Bennett, J., Lanning, S., et al.: The netflix prize. In: Proceedings of KDD cup and workshop. vol. 2007, p. 35. New York, NY, USA. (2007)
4. Chae, D.K., Kang, J.S., Kim, S.W., Lee, J.T.: Cfgan: A generic collaborative filtering framework based on generative adversarial networks. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 137–146. Torino, Italy (Oct 2018)

5. Dai, W., Yang, Q., Xue, G.R., Yu, Y.: Boosting for transfer learning. In: Proceedings of the 24th International Conference on Machine Learning. pp. 193–200. Corvalis, Oregon, USA (Jun 2007)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Gao, S., Luo, H., Chen, D., Li, S., Gallinari, P., Guo, J.: Cross-domain recommendation via cluster-level latent factor model. In: Joint European conference on machine learning and knowledge discovery in databases. pp. 161–176. Prague, Czech (Sep 2013)
8. Gomez-Uribe, C.A., Hunt, N.: The netflix recommender system: Algorithms, business value, and innovation. ACM Trans. Manage. Inf. Syst. **6**(4), 13:1–13:19 (Dec 2015)
9. Han, D., Liu, Q., Fan, W.: A new image classification method using cnn transfer learning and web data augmentation. Expert Systems with Applications **95**, 43 – 56 (2018)
10. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web. pp. 173–182. Perth, Australia (2017)
11. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., Gelly, S.: Parameter-efficient transfer learning for nlp (2019)
12. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification (2018)
13. Hu, G., Zhang, Y., Yang, Q.: Conet: Collaborative cross networks for cross-domain recommendation. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 667–676. Torino, Italy (Oct 2018)
14. Kornblith, S., Shlens, J., Le, Q.V.: Do better imagenet models transfer better? In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2661–2671. CA, US
15. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)
16. Li, B., Yang, Q., Xue, X.: Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In: Twenty-First International Joint Conference on Artificial Intelligence. pp. 2052–2057. CA, USA (Jul 2009)
17. Li, J., Monroe, W., Jurafsky, D.: Understanding neural networks through representation erasure (2016)
18. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. p. 97–105. ICML'15, JMLR.org (2015)
19. Man, T., Shen, H., Jin, X., Cheng, X.: Cross-domain recommendation: An embedding and mapping approach. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17. pp. 2464–2470. Melbourne, Australia (Aug 2017)
20. Ni, J., Li, J., McAuley, J.: Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 188–197. Hong Kong, China (Nov 2019)

21. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2014)
22. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering **22**(10), 1345–1359 (Oct 2010)
23. Pan, W., Xiang, E.W., Liu, N.N., Yang, Q.: Transfer learning in collaborative filtering for sparsity reduction. In: Twenty-fourth AAAI conference on artificial intelligence. pp. 230–235. Georgia, USA (Jul 2010)
24. Ricci, F., Rokach, L., Shapira, B.: Introduction to recommender systems handbook. In: Recommender systems handbook, pp. 1–35. Springer (2011)
25. Sedhain, S., Menon, A.K., Sanner, S., Xie, L.: Autorec: Autoencoders meet collaborative filtering. In: Proceedings of the 24th International Conference on World Wide Web. pp. 111–112. Florence, Italy (May 2015)
26. Seo, S., Huang, J., Yang, H., Liu, Y.: Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In: Proceedings of the Eleventh ACM Conference on Recommender Systems. pp. 297–305. Como, Italy (Aug 2017)
27. Shi, Q., Du, B., Zhang, L.: Domain adaptation for remote sensing image classification: A low-rank reconstruction and instance weighting label propagation inspired algorithm. IEEE Transactions on Geoscience and Remote Sensing **53**(10), 5677–5689 (2015)
28. Smith, B., Linden, G.: Two decades of recommender systems at amazon.com. IEEE Internet Computing **21**(3), 12–18 (May 2017)
29. Voigt, P., Von dem Bussche, A.: The eu general data protection regulation (gdpr). A Practical Guide, 1st Ed., Cham: Springer International Publishing (2017)
30. Wang, Z., Du, B., Guo, Y.: Domain adaptation with neural embedding matching. IEEE Transactions on Neural Networks and Learning Systems pp. 1–11 (2019). https://doi.org/10.1109/TNNLS.2019.2935608
31. Wiese, G., Weissenborn, D., Neves, M.: Neural domain adaptation for biomedical question answering. In: Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017). pp. 281–289. Vancouver, Canada (Aug 2017)
32. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in neural information processing systems. pp. 3320–3328 (2014)
33. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: A survey and new perspectives. ACM Comput. Surv. **52**(1), 5:1–5:38 (Feb 2019)
34. Zhu, F., Chen, C., Wang, Y., Liu, G., Zheng, X.: Dtcdr: A framework for dual-target cross-domain recommendation. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (Dec 2019)
35. Zhu, F., Wang, Y., Chen, C., Liu, G., Orgun, M., Wu, J.: A deep framework for cross-domain and cross-system recommendations. In: IJCAI International Joint Conference on Artificial Intelligence. pp. 3711–3717. Stockholm, Sweden (Jul 2018)
36. Zhu, Z., Wang, J., Caverlee, J.: Improving top-k recommendation via jointcollaborative autoencoders. In: The World Wide Web Conference. WWW '19 (May 2019)

Learning active learning at the crossroads? evaluation and discussion

L. Desreumaux[1], V. Lemaire[2]

[1] SAP Labs, Paris, France
[2] Orange Labs, Lannion, France

Abstract. Active learning aims to reduce annotation cost by predicting which samples are useful for a human expert to label. Although this field is quite old, several important challenges to using active learning in real-world settings still remain unsolved. In particular, most selection strategies are hand-designed, and it has become clear that there is no best active learning strategy that consistently outperforms all others in all applications. This has motivated research into meta-learning algorithms for "learning how to actively learn". In this paper, we compare this kind of approach with the association of a Random Forest with the margin sampling strategy, reported in recent comparative studies as a very competitive heuristic. To this end, we present the results of a benchmark performed on 20 datasets that compares a strategy learned using a recent meta-learning algorithm with margin sampling. We also present some lessons learned and open future perspectives.

1 Introduction

Modern supervised learning methods[3] are known to require large amounts of training examples to reach their full potential. Since these examples are mainly obtained through human experts who manually label samples, the labelling process may have a high cost. Active learning (AL) is a field that includes all the selection strategies that allow to iteratively build the training set of a model in interaction with a human expert, also called oracle. The aim is to select the most informative examples to minimize the labelling cost.

In this article, we consider the selective sampling framework, in which the strategies manipulate a set of examples $\mathcal{D} = \mathcal{L} \cup \mathcal{U}$ of constant size, where $\mathcal{L} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}$ is the set of labelled examples and $\mathcal{U} = \{\boldsymbol{x}_i\}_{i=l+1}^{n}$ is the set of unlabelled examples. In this framework, active learning is an iterative process that continues until a labelling budget is exhausted or a pre-defined performance threshold is reached. Each iteration begins with the selection of the most informative example $\boldsymbol{x}^\star \in \mathcal{U}$. This selection is generally based on information collected during previous iterations (predictions of a classifier, density measures, etc.). The example $\boldsymbol{x}^\star$ is then submitted to the oracle that returns the corresponding class $y^\star$, and the pair $(\boldsymbol{x}^\star, y^\star)$ is added to $\mathcal{L}$. The new learning set is

[3] In this article, we limit ourselves to binary classification problems.

then used to improve the model and the new predictions are used in the next iteration.

The utility measures defined by the active learning strategies in the literature [36] differ in their positioning according to a dilemma between the exploitation of the current classifier and the exploration of the training data. Selecting an unlabelled example in an unknown region of the observation space $\mathbb{R}^d$ helps to explore the data, so as to limit the risk of learning a hypothesis too specific to the current set $\mathcal{L}$. Conversely, selecting an example in a sampled region of $\mathbb{R}^d$ locally refines the predictive model.

1.1 Traditional heuristic-based AL

The active learning field comes from a parallel between active educational methods and machine learning theory. The learner is from now a statistical model and not a student. The interactions between the student and the teacher correspond to the interactions between the model and the oracle. The examples are situations used by the model to generate knowledge on the problem.

The first AL algorithms were designed with the objective of transposing these "educational" methods to the machine learning domain. The easiest way was to keep the usual supervised learning methods and to add "strategies" relying on various heuristics to guide the selection of the most informative examples. From the first initiative and up to now, a lot of strategies motivated by human intuitions have been suggested in the literature. The purpose of this paper is not to give an overview of the existing strategies but the reader may find in [36, 1] of lot of them.

A careful reading of the experimental results published in the literature shows that there is no best AL strategy that consistently outperforms all others in all applications, and some strategies cater to specific classifiers or to specific applications. Based on this observation, several comprehensive benchmarks carried out on numerous datasets have highlighted the strategies which, on average, are the most suitable for several classification models [28, 41, 29]. They are given in Table 1. For example, the most appropriate strategy for logistic regression and random forest is an uncertainty-based sampling[4] strategy, named margin sampling, which consists in selecting at each iteration the instance for which the difference between the probabilities of the two most likely classes is the smallest [34]. To produce this table, we purposefully omitted studies that have a restricted scope, such as focusing on too few datasets [4], specific tasks [37], an insufficient number of strategies [35, 31], or variants of a single strategy [21].

[4] The reader interested in the measures used to quantify the degree of uncertainty in the context of active learning may find in [25, 18] an interesting view which advocates a distinction between two different types of uncertainty, referred to as epistemic and aleatoric.

Strategy	**RF**[1]	**SVM**[2]	**5NN**[3]	**GNB**[4]	**C4.5**[5]	**LR**[6]	**VFDT**[7]
Margin[a]	[29]						
Entropy[b]						[41]	
QBD[c]				[28]			[28]
Density[d]			[29, 28]		[28]		
OER[e]		[29]		[29]	[29]		

Table 1. Best model/strategy associations highlighted in the literature as a guide to use the appropriate strategy versus the classifier. Strategies: (a) Margin sampling, (b) Entropy sampling, (c) Query by Disagreement, (d) Density sampling, (e) Optimistic Error Reduction. Classifiers: (1) Random Forest, (2) Support Vector Machine, (3) 5-Nearest Neighbors, (4) Gaussian Naive Bayes, (5) C4.5 Decision Tree, (6) Logistic Regression, (7) Very Fast Decision Tree.

1.2 Meta-learning approaches to active learning

While the traditional AL strategies can achieve remarkable performance, it is often challenging to predict in advance which strategy is the most suitable in a particular situation. In recent years, meta-learning algorithms have been gaining in popularity [23]. Some of them have been proposed to tackle the problem of learning AL strategies instead of relying on manually designed strategies.

Motivated by the success of methods that combine predictors, the first AL algorithms within this paradigm were designed to combine traditional AL strategies with bandit algorithms [3, 12, 17, 8, 10, 26]. These algorithms learn how to select the best AL criterion for any given dataset and adapt it over time as the learner improves. However, all the learning must be achieved within a few examples to be helpful, and these algorithms suffer from a cold start issue. Moreover, these approaches are restricted to combining existing AL heuristic strategies.

Within the meta-learning framework, some other algorithms have been developed to learn from scratch an AL strategy on multiple source datasets and transfer it to new target datasets [19, 20, 27]. Most of them are based on modern reinforcement learning methods. The key challenge consists in learning an AL strategy that is general enough to automatically control the exploitation/exploration trade-off when used on new unlabelled datasets, which is not possible when using heuristic strategies.

1.3 Objective of this paper

From the state of the art, it appears that meta-learned AL strategies can outperform the most widely used traditional AL strategies, like uncertainty sampling. However, most of the papers that introduce new meta-learning algorithms do not include comprehensive benchmarks that could ascertain the transferability of the learned strategies and demonstrate that these strategies can safely be used in real-world settings.

The objective of this article is thus to compare two possible options in the realization of an AL solution that could be used in an industrial context: using a traditional heuristic-based strategy (see Section 1.1) that, on average, is the best one for a given model and could be used as a strong baseline easy to implement and not so easy to beat, or using a more sophisticated strategy learned in a data-driven fashion that comes from the very recent literature on meta-learning (see Section 1.2).

To this end, we present the results of a benchmark performed on 20 datasets that compares a strategy learned using the meta-learning algorithm proposed in [20] with margin sampling [34], the models used being in both cases logistic regression and random forest. We evaluated the work of [20] since the authors claim to be able to learn a "general-purpose" AL strategy that can generalise across diverse problems and outperform the best heuristic and bandit approaches.

The rest of the paper is organized as follows. In Section 2, we explain all the aspects of the Learning Active Learning (LAL) method proposed in [20], namely the Deep Q-Learning algorithm and the modeling of active learning as a Markov decision process (MDP). In Section 3, we present the protocol used to do extensive comparative experiments on public datasets from various application areas. In Section 4, we give the results of our experimental study and make some observations. Finally, we present some lessons learned and we open future perspectives in Section 5.

2 Learning active learning strategies

2.1 Q-Learning

A Markov decision process is a formalism for modeling the interaction between an agent and its environment. This formalism uses the concepts of *state*, which describes the situation in which the environment finds itself, *action*, which describes the decision made by the agent, and *reward*, received by the agent when it performs an action. The procedure followed by the agent to select the action to be performed at time t is the *policy*. Given a policy π, the *state-action table* is the function $Q^\pi(s,a)$ which gives the expectation of the weighted sum of the rewards received from the state s if the agent first executes the action a and then follows the policy π.

Q-Learning is a reinforcement learning algorithm that estimates the optimal state-action table $Q^\star = \max_\pi Q^\pi$ from interactions between the agent and the environment. The state-action table Q is updated at any time from the current state s, the action $a = \pi(s)$ where π is the policy derived from Q, the reward received r and the next state of the environment s':

$$Q_{t+1}(s,a) = (1-\alpha_t(s,a))Q_t(s,a) + \alpha_t(s,a)\left(r + \gamma \max_{a' \in \mathcal{A}} Q_t(s',a')\right), \qquad (1)$$

where $\gamma \in [0,1[$ is the weighting factor of the rewards and the $\alpha_t(s,a) \in]0,1[$ are the learning steps that determine the weight of the new experience in relation to the knowledge acquired at previous steps. Assuming that all the state-action pairs are visited an infinite number of times and under some conditions on the learning steps, the resulting sequence of state-action tables converges to $Q^\star$ [40].

The goal of a reinforcement learning agent is to maximize the rewards received over the long term. To do this, in addition to actions that seem to lead to high rewards (exploitation), the agent must select potentially suboptimal actions that allow him to acquire new knowledge about the environment (exploration). For Q-Learning, the ϵ-greedy method is the most commonly used to manage this dilemma. It consists in randomly exploring with a probability of ϵ and acting according to a greedy strategy that chooses the best action with a probability of $(1-\epsilon)$. It is also possible to decrease the probability ϵ at each transition to model the fact that exploration becomes less and less useful with time.

2.2 Deep Q-Learning

In the Q-Learning algorithm, if the state-action table is implemented as a two-input table, then it is impossible to deal with high-dimensional problems. It is necessary to use a parametric model that will be noted as $Q(s,a;\boldsymbol{\theta})$. If it is a deep neural network, it is called Deep Q-Learning.

The training of a neural network requires the prior definition of an error criterion to quantify the loss between the value returned by the network and the actual value. In the context of Q-Learning, the latter value does not exist: one can only use the reward obtained after the completion of an action to calculate a new value, and then estimate the error achieved by calculating the difference between the old value and the new one. A possible cost function would thus be the following:

$$L(s,a,r,s',\boldsymbol{\theta}) = \left(r + \gamma \max_{a' \in \mathcal{A}} Q(s',a';\boldsymbol{\theta}) - Q(s,a;\boldsymbol{\theta})\right)^2 . \tag{2}$$

However, this poses an obvious problem: updating the parameters leads to updating the target. In practice, this means that the training procedure does not converge.

In 2013, a successful implementation of Deep Q-Learning introducing several new features was published [24]. The first novelty is the introduction of a target network, which is a copy of the first network that is regularly updated. This has the effect of stabilizing learning. The cost function becomes:

$$L(s,a,r,s',\boldsymbol{\theta},\boldsymbol{\theta}^-) = \left(r + \gamma \max_{a' \in \mathcal{A}} Q(s',a';\boldsymbol{\theta}^-) - Q(s,a;\boldsymbol{\theta})\right)^2 , \tag{3}$$

where $\boldsymbol{\theta}^-$ is the vector of the target network parameters. The second novelty is experience replay. It consists in saving each experience of the agent (s_i, a_i, r_i, s_{i+1}) in a memory of size m and using random samples drawn from it to update the parameters by stochastic gradient descent. This random draw allows to not necessarily select consecutive, potentially correlated experiences.

2.3 Improvements to Deep Q-Learning

Many improvements to Deep Q-Learning have been published since the article that introduced it. We present here the improvements that interest us for the study of the LAL method.

Double Deep Q-Learning. A first improvement is the correction of the overestimation bias. It has indeed been empirically shown that Deep Q-Learning as presented in Section 2.2 can produce a positive bias that increases the convergence time and has a significant negative impact on the quality of the asymptotically obtained policy. The importance of this bias and its consequences have been verified in particular in the configurations the least favourable to its emergence, *i.e.* when the environment and rewards are deterministic. In addition, its value increases with the size of the set of actions. To correct this bias, the solution which has been proposed in [15] consists in not using the parameters $\boldsymbol{\theta}^-$ to both select and evaluate an action. The cost function then becomes:

$$L(s, a, r, s', \boldsymbol{\theta}, \boldsymbol{\theta}^-) = \left(r + \gamma Q\left(s', \arg\max_{a' \in \mathcal{A}} Q(s', a'; \boldsymbol{\theta}); \boldsymbol{\theta}^-\right) - Q(s, a; \boldsymbol{\theta})\right)^2 . \tag{4}$$

Prioritized Experience Replay. Another improvement is the introduction of the notion of priority in experience replay. In its initial version, Deep Q-Learning considers that all the experiences can identically advance learning. However, reusing some experiences at the expense of others can reduce the learning time. This requires the ability to measure the acceleration potential of learning associated with an experience. The priority measure proposed in [33] is the absolute value of the temporal difference error:

$$\delta_i = \left| r_i + \gamma \max_{a' \in \mathcal{A}} Q(s_{i+1}, a'; \boldsymbol{\theta}^-) - Q(s_i, a_i; \boldsymbol{\theta}) \right| . \tag{5}$$

A maximum priority is assigned to each new experience, so that all the experiences are used at least once to update the parameters.

However, the experiences that produce a small temporal difference error at first use may never be reused. To address this issue, a method was introduced in [33] to manage the trade-off between uniform sampling and sampling focusing on experiences producing a large error. It consists in defining the probability of selecting an experience i as follows:

$$p_i = \frac{\rho_i^\beta}{\sum_{k=1}^m \rho_k^\beta}, \quad \text{with} \quad \rho_i = \delta_i + e, \tag{6}$$

where $\beta \in \mathbb{R}^+$ is a parameter that determines the shape of the distribution and e is a small positive constant that guarantees $p_i > 0$. The case where $\beta = 0$ is equivalent to uniform sampling.

2.4 Formulating active learning as a Markov decision process

The formulation of active learning as a MDP is quite natural. In each MDP *state*, the *agent* performs an *action*, which is the selection of an instance to be labelled, and the latter receives a *reward* that depends on the quality of the model learned with the new instance. The active learning strategy becomes the MDP *policy* that associates an action with a state.

In this framework, the iteration t of the policy learning process from a dataset divided into a learning set $\mathcal{D} = \mathcal{L}_t \cup \mathcal{U}_t$ and a test set[5] $\mathcal{D}'$ consists in the following steps:

1. A model $h^{(t)}$ is learned from $\mathcal{L}_t$. Associated with $\mathcal{L}_t$ and $\mathcal{U}_t$, it allows to characterize a state $\boldsymbol{s}_t$.
2. The agent performs the action $\boldsymbol{a}_t = \pi(\boldsymbol{s}_t) \in \mathcal{A}_t$ which defines the instance $\boldsymbol{x}^{(t)} \in \mathcal{U}_t$ to label.
3. The label $y^{(t)}$ associated with $\boldsymbol{x}^{(t)}$ is retrieved and the training set is updated, *i.e.* $\mathcal{L}_{t+1} = \mathcal{L}_t \cup \{(\boldsymbol{x}^{(t)}, y^{(t)})\}$ and $\mathcal{U}_{t+1} = \mathcal{U}_t \setminus \{\boldsymbol{x}^{(t)}\}$.
4. The agent receives the reward r_t associated with the performance ℓ_t on the test set $\mathcal{D}'$. This reward is used to update the policy (see Section 2.5).

The set of actions $\mathcal{A}_t$ depends on time because it is not possible to select the same instance several times. These steps are repeated until a terminal state s_T is reached. Here, we consider that we are in a terminal state when all the instances have been labelled or when $\ell_t \geq q$, where q is a performance threshold that has been chosen as 98% of the performance obtained when the model is learned on all the training data.

The precise definition of the set of states, the set of actions and the reward function is not evident. To define a state, it has been proposed to use a vector whose components are the scores $\widehat{y}_t(\boldsymbol{x}) = \mathbb{P}(\mathrm{Y} = 0 \,|\, \boldsymbol{x})$ associated with the unlabelled instances of a subset $\mathcal{V}$ set aside. This is the simplest representation that can be used to characterize the uncertainty of a classifier on a dataset at a given time t.

The set of actions has been defined at iteration t as the set of vectors $\boldsymbol{a}_i = [\widehat{y}_t(\boldsymbol{x}_i), g(\boldsymbol{x}_i, \mathcal{L}_t), g(\boldsymbol{x}_i, \mathcal{U}_t)]$, where $\boldsymbol{x}_i \in \mathcal{U}_t$ and :

$$g(\boldsymbol{x}_i, \mathcal{L}_t) = \frac{1}{|\mathcal{L}_t|} \sum_{\boldsymbol{x}_j \in \mathcal{L}_t} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad g(\boldsymbol{x}_i, \mathcal{U}_t) = \frac{1}{|\mathcal{U}_t|} \sum_{\boldsymbol{x}_j \in \mathcal{U}_t} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{7}$$

where dist is the cosine distance. An action is therefore characterized by the uncertainty on the associated instance, as well as by two statistics related to the density of the neighbourhood of the instance.

The reward function has been chosen constant and negative until arrival in a terminal state ($r_t = -1$). Thus, to maximize its reward, the agent must perform as few interactions as possible.

[5] Given that active learning is usually applied in cases, this test set assumed to be small or very small the performance evaluated on this test set could be a possibly bad approximation. This issue and techniques for avoiding it are not examined in this paper.

2.5 Learning the optimal policy through Deep Q-Learning

The Deep Q-Learning algorithm with the improvements presented in Section 2.3 is used to learn the optimal policy. To be able to process a state space that evolves with each iteration, the neural network architecture has been modified. The new architecture considers actions as inputs to the Q function in the same way as states. It then returns only one value, while the classical architecture takes only one state as input and returns the values associated with all the actions.

The learning procedure involves a collection of Z labelled datasets $\{\mathcal{Z}_i\}_{1\leq i \leq Z}$. It consists in repeating the following steps (see Figure 1):

1. A dataset $\mathcal{Z} \in \{\mathcal{Z}_i\}$ is randomly selected and divided into a training set $\mathcal{D}$ and a test set $\mathcal{D}'$.
2. The policy π derived from the Deep Q-Network is used to simulate several active learning episodes on $\mathcal{Z}$ according to the procedure described in Section 2.4. Experiences $(\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1})$ are collected in a finite size memory.
3. The Deep Q-Network parameters are updated several times from a mini-batch of experiences extracted from the memory (according to the method described in Section 2.3).

To initialize the Deep Q-Network, some warm start episodes are simulated using a random sampling policy, followed by several parameter updates. Once the strategy is learned, its deployment is very simple. At each iteration of the sampling process, the classifier is re-trained, then the vector characterizing the process state and all the vectors associated with the actions are calculated. The vector $\boldsymbol{a}^\star$ corresponding to the example to label $\boldsymbol{x}^\star$ is then the one that satisfies $\boldsymbol{a}^\star = \arg\max_{\boldsymbol{a}\in\mathcal{A}} Q(\boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta})$, the parameters $\boldsymbol{\theta}$ being set at the end of the policy learning procedure.

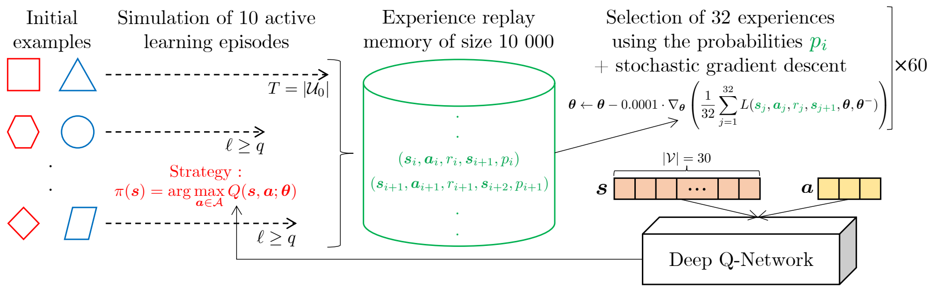

Initial examples
Simulation of 10 active learning episodes
Experience replay memory of size 10 000
Selection of 32 experiences using the probabilities p_i + stochastic gradient descent
×60
$T = |\mathcal{U}_0|$
$\ell \geq q$
Strategy : $\pi(\boldsymbol{s}) = \arg\max_{\boldsymbol{a}\in\mathcal{A}} Q(\boldsymbol{s},\boldsymbol{a};\boldsymbol{\theta})$
$\ell \geq q$
$(\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_{i+1}, p_i)$
$(\boldsymbol{s}_{i+1}, \boldsymbol{a}_{i+1}, r_{i+1}, \boldsymbol{s}_{i+2}, p_{i+1})$
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - 0.0001 \cdot \nabla_{\boldsymbol{\theta}} \left(\frac{1}{32} \sum_{j=1}^{32} L(\boldsymbol{s}_j, \boldsymbol{a}_j, r_j, \boldsymbol{s}_{j+1}, \boldsymbol{\theta}, \boldsymbol{\theta}^-) \right)$
$|\mathcal{V}| = 30$
$\boldsymbol{s}$
$\boldsymbol{a}$
Deep Q-Network

Fig. 1. Illustration of the different steps involved in an iteration of the policy learning phase using Deep Q-Learning (the arrows give intuitions about main steps and data flows)

.

3 Experimental protocol

In this section, we introduce our protocol of the comparative experimental study we conducted.

3.1 Policy learning

To learn the strategy, we used the same code[6], the same hyperparameters and the same datasets as those used in [20]. The complete list of hyperparameters is given in Table 2 with the variable names from the code that represent them. The datasets from which the strategy is learned are given in Table 3.

The specification of the neural network architecture is very simple (all the layers are fully connected): (i) the first layer (linear + sigmoid) receives the vector $\boldsymbol{s}$ (*i.e.* $|\mathcal{V}| = 30$ input neurons) and has 10 output neurons; (ii) the second layer (linear + sigmoid) concatenates the 10 output neurons of the first layer with the vector $\boldsymbol{a}$ (*i.e.* 13 neurons in total) and has 5 output neurons; (iii) finally, the last layer (linear) has only one output to estimate $Q(\boldsymbol{s}, \boldsymbol{a})$.

Hyperparameter	Description
`N_STATE_ESTIMATION = 30`	Size of $\mathcal{V}$
`REPLAY_BUFFER_SIZE = 10000`	Experience replay memory size
`PRIORITIZED_REPLAY_EXPONENT = 3`	Exponent β involved in Equation (6)
`BATCH_SIZE = 32`	Minibatch size for stochastic gradient descent
`LEARNING_RATE = 0.0001`	Learning rate
`TARGET_COPY_FACTOR = 0.01`	Value that sets the target network update[1]
`EPSILON_START = 1`	Exploration probability at start
`EPSILON_END = 0.1`	Minimum exploration probability
`EPSILON_STEPS = 1000`	Number of updates of ϵ during the training
`WARM_START_EPISODES = 100`	Number of warm start episodes
`NN_UPDATES_PER_WARM_START = 100`	Number of parameter updates after the warm start
`TRAINING_ITERATIONS = 1000`	Number of training iterations
`TRAINING_EPISODES_PER_ITERATION = 10`	Number of episodes per training iteration
`NN_UPDATES_PER_ITERATION = 60`	Number of updates per training iteration

[1] In this implementation, the target network parameters $\boldsymbol{\theta}^-$ are updated each time the parameters $\boldsymbol{\theta}$ are changed as follows: $\boldsymbol{\theta}^- \leftarrow (1-\texttt{TARGET\_COPY\_FACTOR})\cdot\boldsymbol{\theta}^- + \texttt{TARGET\_COPY\_FACTOR}\cdot\boldsymbol{\theta}$.

Table 2. Hyperparameters involved in Deep Q-Learning.

3.2 Traditional heuristic-based AL used as baseline: margin sampling

Our objective is to compare the performance of a strategy learned using LAL with the performance of a heuristic strategy that, on average, is the best one for

[6] `https://github.com/ksenia-konyushkova/LAL-RL`

Dataset	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{Y}\rvert$	#num	#cat	maj (%)	min (%)
australian	690	2	6	8	55.51	44.49
breast-cancer	272	2	0	9	70.22	29.78
diabetes	768	2	8	0	65.10	34.90
german	1000	2	7	13	70.00	30.00
heart	293	2	13	0	63.82	36.18
ionosphere	350	2	33	0	64.29	35.71
mushroom	8124	2	0	21	51.80	48.20
wdbc	569	2	30	0	62.74	37.26

Table 3. Datasets used to learn the new strategy. Columns: number of examples, number of classes, numbers of numerical and categorical variables, proportions of examples in the majority and minority classes.

a given model. Several benchmarks conducted on numerous datasets have highlighted the fact that margin sampling is the best heuristic strategy for logistic regression (LR) and random forest (RF) [41, 29].

Margin sampling consists in choosing the instance for which the difference (or margin) between the probabilities of the two most likely classes is the smallest:

$$\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x}\in\mathcal{U}} \mathbb{P}(y_1 \mid \boldsymbol{x}) - \mathbb{P}(y_2 \mid \boldsymbol{x}), \tag{8}$$

where y_1 and y_2 are respectively the first and second most probable classes for $\boldsymbol{x}$. The main advantage of this strategy is that it is easy to implement: at each iteration, a single training of the model and $\lvert\mathcal{U}\rvert$ predictions are sufficient to select an example to label. A major disadvantage, however, is its total lack of exploration, as it only exploits locally the hypothesis learned by the model.

We chose to evaluate the Margin/LR association because it is with logistic regression that the hyperparameters of Table 2 were optimized in [20]. In addition, in order to determine whether it is necessary to modify them when another model is used, we also evaluated the Margin/RF association. This last association is particularly interesting because it is the best association highlighted in a recent and large benchmark carried out on 73 datasets, including 5 classification models and 8 active learning strategies [29]. In addition, we evaluated random sampling (Rnd) for both models.

3.3 Datasets

The datasets were selected so as to have a high diversity according to the following criteria: (i) number of examples; (ii) number of numerical variables; (iii) number of categorical variables; (iv) class imbalance.

We have also taken care to exclude datasets that are too small and not representative of those used in an industrial context. The 20 selected datasets are described in Table 4. They all come from the UCI database [11], apart from the dataset "orange-fraud" which is dataset on fraud detection. Four of the datasets have been used in a challenge on active learning that took place in 2010 [14] and the dataset "nomao" comes from another challenge on active learning [6].

Dataset	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{Y}\rvert$	**#num**	**#cat**	**maj (%)**	**min (%)**
adult	48790	2	6	8	76.06	23.94
banana	5292	2	2	0	55.16	44.84
bank-marketing-full	45211	2	7	9	88.30	11.70
climate-simulation-craches	540	2	20	0	91.48	8.52
eeg-eye-state	14980	2	14	0	55.12	44.88
hiva	40764	2	1617	0	96.50	3.50
ibn-sina	13951	2	92	0	76.18	23.82
magic	18905	2	10	0	65.23	34.77
musk	6581	2	166	1	84.55	15.45
nomao	32062	2	89	29	69.40	30.60
orange-fraud	1680	2	16	0	63.75	36.25
ozone-onehr	2528	2	72	0	97.11	2.89
qsar-biodegradation	1052	2	41	0	66.35	33.65
seismic-bumps	2578	2	14	4	93.41	6.59
skin-segmentation	51444	2	3	0	71.51	28.49
statlog-german-credit	1000	2	7	13	70.00	30.00
thoracic-surgery	470	2	3	13	85.11	14.89
thyroid-hypothyroid	3086	2	7	18	95.43	4.57
wilt	4819	2	5	0	94.67	5.33
zebra	61488	2	154	0	95.42	4.58

Table 4. Datasets used for the evaluation of the strategy learned by LAL. Columns: number of examples, number of classes, numbers of numerical and categorical variables, proportions of examples in the majority and minority classes.

3.4 Evaluation criteria

In our evaluation protocol, the active sampling process begins with the random selection of one instance in each class and ends when 250 instances are labelled. This value ensures that our results are comparable to other studies in the literature. For performance comparison, we used the area under the learning curve (ALC) based on the classification accuracy. We do not claim that the ALC is a "perfect metric"[7] but it is the defacto standard evaluation criterion in active learning, and it has been chosen as part of a challenge [14].

Our evaluation was carried out by cross-validation with 5 partitions, in which class imbalance within the complete dataset was preserved. For each partition, the sampling process was repeated 5 times with different initializations to get a

[7] There is literature on more expressive summary statistics of the active-learning curve [39, 30]. This could be a limitation of this current article, other metrics could be tester in future versions of experiments.

mean and a variance on the result. However, we have made sure that the initial instances are identical for all the strategy/model associations on each partition so as to not introduce bias into the results. In addition, for Rnd, the random sequence of numbers was identical for all the models.

4 Results

The results of our experimental study are given in Table 5. The mean ALC obtained for each dataset/classifier/strategy association are reported (the optimal score is 100). The left part of the table gives the results for logistic regression and the right part gives the results for random forest. The penultimate line corresponds to the averages calculated on all the datasets and the last line gives the number of times the strategy has won, tied or lost. The non-significant differences were established on the basis of a paired t-test at 99% significance level (where H0: same mean between populations and where the mean is the estimate out of 5 repetitions x cross-validation with 5 partitions of each method).

Dataset	**Rnd/LR**	**Margin/LR**	**LAL/LR**	**Rnd/RF**	**Margin/RF**	**LAL/RF**	**maj**
adult	77.93	78.91	78.97	80.17	**81.27**	**81.21**	76.06
banana	53.03	**57.39**	53.12	**80.24**	73.81	73.58	55.16
bank-marketing-full	86.85	**87.62**	**87.72**	88.19	88.34	88.49	88.30
climate-simulation	87.22	**89.13**	**88.62**	91.15	91.14	91.13	91.48
eeg-eye-state	56.08	55.32	56.11	65.53	**67.58**	64.42	55.12
hiva	64.43	**70.84**	**71.80**	96.32	**96.47**	**96.44**	96.50
ibn-sina	84.77	**88.58**	**88.90**	90.53	**93.41**	**92.75**	76.18
magic	76.49	**77.93**	**77.64**	78.05	**80.79**	**79.68**	65.23
musk	83.73	82.34	81.95	89.55	**96.18**	95.35	84.55
nomao	89.45	**91.43**	**91.37**	89.41	**92.32**	**92.07**	69.40
orange-fraud	76.70	**81.74**	74.26	89.15	**90.66**	**90.48**	63.75
ozone-onehr	92.90	94.26	**95.06**	96.61	96.83	96.89	97.11
qsar-biodegradation	80.98	**82.62**	**83.53**	80.34	**82.76**	**82.40**	66.35
seismic-bumps	90.87	**92.59**	**92.14**	92.48	**92.92**	**93.02**	93.41
skin-segmentation	77.05	**82.69**	**83.21**	91.51	**95.70**	**95.77**	71.51
statlog-german-credit	70.76	**72.12**	**72.34**	72.25	72.93	72.78	70.00
thoracic-surgery	83.76	83.93	82.72	83.51	**84.41**	**84.18**	85.11
thyroid-hypothyroid	97.21	**97.99**	**97.97**	97.75	**98.77**	**98.71**	95.43
wilt	93.53	**95.18**	92.87	94.86	**97.23**	**97.02**	94.67
zebra	86.40	90.31	**91.36**	94.71	**95.54**	95.25	95.42
Mean	80.51	**82.65**	82.08	87.12	**88.45**	88.08	79.53
win/tie/loss	0/5/15	**3/15/2**	2/15/3	1/4/15	**3/16/1**	0/16/4	

Table 5. Results of the experimental study.

Several observations can be made. First of all, it should be noted that the choice of model is decisive: the results of random forest are all better than those of logistic regression. The random forest model learns indeed very well from few data, as highlighted in [32]. We can notice that even with random sampling, RF is almost always better than LR, regardless of the strategy used. In addition, using margin sampling with this model allows a significant performance improvement. This model is very competitive in itself because by its nature, it includes terms of exploration and exploitation (see Section 5 Conclusion about this point).

In addition, the results of the learned strategy clearly show that a good active learning strategy has been learned, since it performs better than random sampling over a large number of datasets. However, the learned strategy is no better than margin sampling. These results are nevertheless very interesting since only 8 datasets were used in the learning procedure.

Finally, the results show a well-known fact about active learning: on very unbalanced datasets, it is difficult to achieve a better performance than random sampling, as shown in the last column of Table 5 in which the results obtained by always predicting the majority class are given. The "cold start" problem that occurs in active learning, *i.e.* the inability of making reliable predictions in early iterations (when training data is not sufficient), is indeed further aggravated when a dataset has highly imbalanced classes, since the selected samples are likely to belong to the majority class [38]. However, if the imbalance is known, it may be interesting to associate strategies with a model or criterion appropriate to this case, as illustrated in [13].

To investigate the "learning speed", we show results for different sizes of $\mathcal{L}$ in Table 6. They lead to similar conclusions and our results for $|\mathcal{L}| = 32$ confirm the results of [32]. The reader may find all our experimental results on Github[8].

	$\|\mathcal{L}\| = 32$			$\|\mathcal{L}\| = 64$			$\|\mathcal{L}\| = 128$			$\|\mathcal{L}\| = 250$		
Dataset	**Rnd**	**Margin**	**LAL**	**Rnd**	**Margin**	**LAL**	**Rnd**	**Margin**	**LAL**	**Rnd**	**Margin**	**LAL**
adult	77.95	77.88	78.16	79.72	80.51	81.05	81.13	82.79	82.48	82.12	83.55	83.40
banana	71.13	65.48	65.16	77.93	71.42	70.96	83.64	75.58	75.70	86.55	79.71	81.35
bank...	88.05	87.90	88.10	88.29	88.38	88.54	88.43	88.82	88.90	88.75	89.21	89.35
climate...	91.26	91.26	91.18	91.40	91.29	91.40	91.26	91.33	91.33	91.44	91.22	91.29
eeg...	58.28	58.94	57.34	62.07	63.17	60.79	66.77	69.38	65.35	72.55	75.08	72.46
hiva	96.36	96.52	96.49	96.36	96.55	96.54	96.46	96.57	96.56	96.49	96.65	96.65
ibn-sina	86.88	91.17	89.78	90.48	93.99	92.96	92.73	94.76	94.25	93.86	95.85	95.48
magic	71.99	75.63	72.95	76.85	80.20	77.26	80.15	82.71	82.01	82.42	84.53	84.43
musk	85.29	89.50	90.09	87.43	94.44	94.18	90.58	98.78	97.63	93.64	99.98	99.31
nomao	85.92	89.35	89.37	88.92	92.46	92.09	90.85	93.69	93.33	92.36	94.52	94.37
orange...	88.06	90.36	90.09	89.16	90.98	90.67	90.08	91.72	91.33	90.41	91.85	91.74
ozone...	96.36	96.97	97.01	96.74	97.04	97.10	96.93	97.08	97.11	97.02	97.03	97.05
qsar...	75.75	78.08	76.61	79.75	82.09	81.42	81.94	84.65	84.88	84.03	86.12	86.08
seismic...	92.39	93.21	93.19	92.42	93.28	93.19	92.52	93.26	93.20	93.14	93.08	93.28
skin...	86.42	89.19	89.46	90.80	96.19	96.06	93.70	98.86	98.65	95.85	99.56	99.49
statlog...	70.36	70.70	69.70	70.94	72.47	71.75	72.40	73.46	74.10	74.29	75.22	75.06
thoracic...	83.14	84.42	84.12	83.31	85.02	84.76	83.70	84.89	84.68	84.21	84.51	84.68
thyroid...	97.26	98.71	98.43	97.86	99.15	98.71	98.08	99.10	98.89	98.26	98.84	98.98
wilt	94.60	96.23	95.98	95.01	97.47	96.90	95.30	98.21	97.64	96.07	98.51	98.37
zebra	94.66	95.32	95.28	94.87	95.44	95.31	94.96	95.72	95.46	95.01	96.04	95.33
Mean	84.60	**85.84**	85.42	86.51	**88.07**	87.58	88.08	**89.56**	89.17	89.42	**90.55**	90.40

Table 6. Mean test accuracy (%) for different sizes of $|\mathcal{L}|$ with the random forest model.

[8] `https://github.com/ldesreumaux/lal_evaluation`

5 Discussion and open questions

In this article, we evaluated a method representative of a recent orientation of active learning research towards meta-learning methods for “learning how to actively learn”, which is on top of the state of the art [20], versus a traditional heuristic-based Active Learning (the association of Random Forest and Margin) which is one of the best method reported in recent comparative studies [41, 29]. The comparison is limited to just one representative of each of the two classes (meta-learning and traditional heuristic-based) but since each is on top of the state of the art several lessons can be drawn from our study.

Relevance of LAL. First of all, the experiments carried out confirm the relevance of the LAL method, since it has enabled us to learn a strategy that achieves the performance of a very good heuristic, namely margin sampling, but contrary to the results in [20], the strategy is not always better than random sampling. This method still raises many problems, including that of the transferability of the learned strategies. An active learning solution that can be used in an industrial context must perform well on real data of an unknown nature and must not involve parameters to be adjusted. With regard to the LAL method, a first major problem is therefore the constitution of a “dataset of datasets” large and varied enough to learn a strategy that is effective in very different contexts.

Moreover, the learning procedure is sensitive to the performance criteria used, which in our view seems to be a problem. Ideally, the strategy learned should be usable on new datasets with arbitrary performance criteria (AUC, F-score, etc.). From our point of view, the work of optimizing the many hyperparameters of the method (see Table 2) can not be carried out by a user with no expertise in deep reinforcement learning.

About the Margin/RF association. In addition to the evaluation of the LAL method, we confirmed a result of [29], namely that margin sampling, associated with a random forest, is a very competitive strategy. From an industrial point of view, regarding the computational complexity, the performances obtained and the absence of “domain knowledge required to be used” the Margin/RF association remains a very strong baseline difficult to beat. However, it shares a major drawback with many active learning strategies, that is its lack of reliability. Indeed, there is no strategy that is better or equivalent to random sampling on **all** datasets and with all models. The literature on active learning is incomplete with regard to this problem, which is nevertheless a major obstacle to using active learning in real-world settings.

Another important problem in real-world applications, little studied in the literature, is the estimation of the generalization error without a test set. It would be interesting to check if the Out-Of-Bag samples of the random forests [5] can be used in an active learning context to estimate this error.

Concerning the exploitation/exploration dilemma, margin sampling clearly performs only exploitation. The good results of the Margin/RF association may suggest that the RF algorithm intrinsically contains a part of exploration due to the bagging paradigm. It could be interesting to add experiments in the future to test this point.

Still with regard to the random forests, an open question is to study if a better strategy than margin sampling could be designed. Since the random forests are ensemble classifiers, a possible way of research to design this strategy is to check if they could be used in the credal uncertainty framework [2] which seeks to differentiate between the reducible and irreducible part of the uncertainty in a prediction.

About error generalization. In Real world application AL should be used most of the time in absence of a test dataset. A open question could be to a use another known result about RF: the possibility to have an estimate of the generalization error using the Out-Of-Bag (OOB) samples [16, 5]. We did not present experiments on this topic in this paper but an idea could be to analyze the convergence versus the number of labelled examples between the OOB performance and the test performance to check at which "moment" ($|L|$) one could trust[9] the OOB performance (OOB performance $\approx$ test performance). The use of a "random uniform forest" [9] for which the OOB performance seems to be more reliable could also be investigated.

About the benchmarking methodology. Recent benchmarks have highlighted the need for extensive experimentation to compare active learning strategies. The research community might benefit from a "reference" benchmark, as in the field of time series classification [7], so that new results can be rigorously compared to the state of the art on a same and large set of datasets. By this way, one will have comprehensive benchmarks that could ascertain the transferability of the learned strategies and demonstrate that these strategies can safely be used in real-world settings.

If this reference benchmark is created, the second step would be to decide how to compare the AL strategies. This comparison could be made using not a single criterion but a "pool" of criteria. This pool may be chosen to reflect different "aspects" of the results [22].

References

1. Aggarwal, C.C., Kong, X., Gu, Q., Han, J., Yu, P.S.: Active Learning: A Survey. In: Aggarwal, C.C. (ed.) Data Classification: Algorithms and Applications, chap. 22, pp. 571–605. CRC Press (2014)
2. Antonucci, A., Corani, G., Bernaschina, S.: Active Learning by the Naive Credal Classifier. In: Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM). pp. 3–10 (2012)
3. Baram, Y., El-Yaniv, R., Luz, K.: Online Choice of Active Learning Algorithms. Journal of Machine Learning Research **5**, 255–291 (2004)

[9] Since when $|L|$ is very low the RF do overtraining thus it's train performance is not a good indicator for the error generalization

4. Beyer, C., Krempl, G., Lemaire, V.: How to Select Information That Matters: A Comparative Study on Active Learning Strategies for Classification. In: Proceedings of the 15th International Conference on Knowledge Technologies and Data-driven Business. ACM (2015)
5. Breiman, L.: Out-of-bag estimation (1996), `https://www.stat.berkeley.edu/~breiman/OOBestimation.pdf`, last visited 08/03/2020
6. Candillier, L., Lemaire, V.: Design and analysis of the nomao challenge active learning in the real-world. In: Proceedings of the ALRA: Active Learning in Real-world Applications, Workshop ECML-PKDD. (2012)
7. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The UCR Time Series Classification Archive (2015), `www.cs.ucr.edu/~eamonn/time_series_data/`
8. Chu, H.M., Lin, H.T.: Can Active Learning Experience Be Transferred? 2016 IEEE 16th International Conference on Data Mining pp. 841–846 (2016)
9. Ciss, S.: Generalization Error and Out-of-bag Bounds in Random (Uniform) Forests, working paper or preprint, `https://hal.archives-ouvertes.fr/hal-01110524/document`, last visited 06/03/2020
10. Collet, T.: Optimistic Methods in Active Learning for Classification. Ph.D. thesis, Université de Lorraine (2018)
11. Dua, D., Graff, C.: UCI Machine Learning Repository (2017), `http://archive.ics.uci.edu/ml`
12. Ebert, S., Fritz, M., Schiele, B.: Ralf: A reinforced active learning formulation for object class recognition. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3626–3633 (2012)
13. Ertekin, S., Huang, J., Bottou, L., Giles, L.: Learning on the Border: Active Learning in Imbalanced Data Classification. In: Conference on Information and Knowledge Management. pp. 127–136. CIKM (2007)
14. Guyon, I., Cawley, G., Dror, G., Lemaire, V.: Results of the Active Learning Challenge. In: Proceedings of Machine Learning Research. vol. 16, pp. 19–45. PMLR (2011)
15. Hasselt, H.v., Guez, A., Silver, D.: Deep Reinforcement Learning with Double Q-Learning. In: AAAI Conference on Artificial Intelligence. pp. 2094–2100 (2016)
16. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: data mining, inference and prediction. Springer, 2 edn. (2009)
17. Hsu, W.N., Lin, H.T.: Active Learning by Learning. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. pp. 2659–2665. AAAI Press (2015)
18. Hüllermeier, E., Waegeman, W.: Aleatoric and Epistemic Uncertainty in Machine Learning: An Introduction to Concepts and Methods. arXiv:1910.09457 [cs.LG] (2019)
19. Konyushkova, K., Sznitman, R., Fua, P.: Learning Active Learning from Data. In: Advances in Neural Information Processing Systems 30, pp. 4225–4235 (2017)
20. Konyushkova, K., Sznitman, R., Fua, P.: Discovering General-Purpose Active Learning Strategies. arXiv:1810.04114 [cs.LG] (2019)
21. Körner, C., Wrobel, S.: Multi-class Ensemble-Based Active Learning. In: Proceedings of the 17th European Conference on Machine Learning. pp. 687–694. Springer-Verlag (2006)
22. Kottke, D., Calma, A., Huseljic, D., Krempl, G., 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)

23. Lemke, C., Budka, M., Gabrys, B.: Metalearning: a survey of trends and technologies. Artificial Intelligence Review **44**, 117–130 (2015)
24. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing Atari with Deep Reinforcement Learning. arXiv:1312.5602 [cs.LG] (2013)
25. Nguyen, V.L., Destercke, S., Hüllermeier, E.: Epistemic Uncertainty Sampling. In: Discovery Science (2019)
26. Pang, K., Dong, M., Wu, Y., Hospedales, T.M.: Dynamic Ensemble Active Learning: A Non-Stationary Bandit with Expert Advice. In: Proceedings of the 24th International Conference on Pattern Recognition. pp. 2269–2276 (2018)
27. Pang, K., Dong, M., Wu, Y., Hospedales, T.M.: Meta-Learning Transferable Active Learning Policies by Deep Reinforcement Learning. arXiv:1806.04798 [cs.LG] (2018)
28. Pereira-Santos, D., de Carvalho, A.C.: Comparison of Active Learning Strategies and Proposal of a Multiclass Hypothesis Space Search. In: Proceedings of the 9th International Conference on Hybrid Artificial Intelligence Systems – Volume 8480. pp. 618–629. Springer-Verlag (2014)
29. Pereira-Santos, D., Prudêncio, R.B.C., de Carvalho, A.C.: Empirical investigation of active learning strategies. Neurocomputing **326–327**, 15–27 (2019)
30. Pupo, O.G.R., Altalhi, A.H., Ventura, S.: Statistical comparisons of active learning strategies over multiple datasets. Knowl. Based Syst. **145**, 274–288 (2018)
31. Ramirez-Loaiza, M.E., Sharma, M., Kumar, G., Bilgic, M.: Active learning: an empirical study of common baselines. Data Mining and Knowledge Discovery **31**(2), 287–313 (2017)
32. Salperwyck, C., Lemaire, V.: Learning with few examples: an empirical study on leading classifiers. In: Proceedings of the 2011 International Joint Conference on Neural Networks. pp. 1010–1019. IEEE (2011)
33. Schaul, T., Quan, J., Antonoglou, I., Silver, D.: Prioritized Experience Replay. arXiv:1511.05952 [cs.LG] (2016)
34. Scheffer, T., Decomain, C., Wrobel, S.: Active Hidden Markov Models for Information Extraction. In: Hoffmann, F., Hand, D.J., Adams, N., Fisher, D., Guimaraes, G. (eds.) Advances in Intelligent Data Analysis. pp. 309–318 (2001)
35. Schein, A.I., Ungar, L.H.: Active learning for logistic regression: an evaluation. Machine Learning **68**, 235–265 (2007)
36. Settles, B.: Active Learning. Morgan & Claypool Publishers (2012)
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. Shao, J., Wang, Q., Liu, F.: Learning to Sample: An Active Learning Framework. IEEE International Conference on Data Mining (ICDM) pp. 538–547 (2019)
39. Trittenbach, H., Englhardt, A., Böhm, K.: An overview and a benchmark of active learning for one-class classification. CoRR **abs/1808.04759** (2018)
40. Watkins, C.J.C.H., Dayan, P.: Q-learning. Machine Learning **8**(3), 279–292 (1992)
41. Yang, Y., Loog, M.: A benchmark and comparison of active learning for logistic regression. Pattern Recognition **83**, 401–415 (2018)

The Effects of Reluctant and Fallible Users in Interactive Online Machine Learning

Agnes Tegen, Paul Davidsson, and Jan A. Persson

Internet of Things and People Research Center, Department of Computer Science and Media Technology, Malmö University, Malmö, Sweden agnes.tegen@mau.se

Abstract. In interactive machine learning it is important to select the most informative data instances to label in order to minimize the effort of the human user. There are basically two categories of interactive machine learning. In the first category, active learning, it is the computational learner that selects which data to be labelled by the human user, whereas in the second one, machine teaching, the selection is done by the human teacher. It is often assumed that the human user is a perfect oracle, i.e., a label will always be provided in accordance with the interactive learning strategy and that this label will always be correct. In real-world scenarios however, these assumptions typically do not hold. In this work, we investigate how the reliability of the user providing labels affects the performance of online machine learning. Specifically, we study reluctance, i.e., to what extent the user does not provide labels in accordance with the strategy, and fallibility, i.e., to what extent the provided labels are incorrect. We show results of experiments on a benchmark dataset as well as a synthetically created dataset. By varying the degree of reluctance and fallibility of the user, the robustness of the different interactive learning strategies and machine learning algorithms is explored. The experiments show that there is a varying robustness of the strategies and algorithms. Moreover, certain machine learning algorithms are more robust towards reluctance compared to fallibility, while the opposite is true for others.

1 Introduction

Active learning [13] and machine teaching [17,18] are two different categories of interactive machine learning strategies that can be used to decrease the amount of labelled data needed to train a machine learning algorithm, while still preserving a high performance. Labelling data is often costly and demands a lot of work from a human user that is meant to provide the labels. In interactive machine learning, a smaller selection of the instances are instead chosen for labelling, where the size of the selection is decided by a labelling budget. The aim of the interactive learning strategy is to pick the instances that will provide most information to the machine learning algorithm.

Interactive online learning is a special case of interactive learning, where the data arrives in a single-pass streaming fashion and each data instance can only be processed by the interactive learning strategy when it arrives. Thus, a decision has to be made at each point in time whether a label should be provided or not for the current data instance.

Generally in interactive learning, the assumptions of the user are that they will always provide a label when queried by an active learning strategy or in accordance with an machine teaching strategy. Furthermore, it is typically assumed that the label provided by the user always is correct. In some settings this can be reasonable assumptions, for instance a medical doctor labelling patient data within their expertise, but in many scenarios they do not hold. While the intent of the assumptions might be to create simplifications in experiments based on complex real-world settings, they may result in conclusions that are not valid. For instance, in an idealised setting, where the user always responds with a correct label, one approach might give the best performance compared to an alternative approach. This does not necessarily mean that the approach will still be the highest performing one if not all labels are correct. If the idealised setting is a simplification of the real setting and the user sometimes does provide an incorrect label, the alternative approach might be the better choice.

In this work we explore how the reliability of a user providing labels affects the performance of online machine learning. We look at the aspects of reluctance, how probable is it that a user will not provide a label in accordance with a given interactive learning strategy, and fallibility, how probable is it that the label provided by a user is incorrect. By varying the degree of reluctance and fallibility of the user we study how this influence the performance of different interactive online machine learning strategies.

2 Related work

In most work on interactive machine learning, the assumptions are that the user will always provide a correct label when queried by an active learning strategy or in accordance with a machine teaching strategy. Furthermore, when the assumptions are made, it is rarely discussed whether they are realistic for the given scenario. In previous work that explore settings where the assumptions are relaxed, this type of user is often referred to as an imperfect oracle. The term is in contrast to the standard definition of oracle in active learning, which is always assumed to respond to a query with a correct label.

Yan et al. explores an active learning setting where the user might return incorrect labels, but might also abstain from labelling [15]. The results show that learning with a user that might abstain is easier than a user that might provide incorrect labels, as an abstention response never mislead the learning algorithm, unlike incorrect labels. However, in this setting the learner can request the label of any data point in the instance space. In our setting the instances are presented in a single-pass streaming manner.

Bouguelia et al. introduces an active learning strategy that handles incorrectly labelled instances, without relying on crowdsourcing [3]. Experiments compare the strategy to multiple benchmark strategies and showcase that the proposed strategy achieves better performance than several of them. The experimental setup is not single-pass and does not address the cold start problem.

The effect of feature noise in an active learning setting is studied by Ramdas et al. [12]. They conclude that active learning results in better performance compared to passive learning even with the presence of feature noise.

Miu et al. present an online active learning framework [10]. Annotations provided by a user is collected in real-time and used for Human Activity Recognition tasks. Apart from testing the proposed framework on benchmark datasets, it was also tested in user studies, by implementing it in a mobile app through which participants could provide labels. In the user study, the replies from the user were simulated to be incorrect 10 % of the time. Apart from baselines, only one interactive learning strategy and one machine learning algorithm was used in the experiments.

Shickel et al. also introduces a framework for active learning in an online setting with multiple imperfect oracles [14]. The framework can query multiple different oracles, based on when they are available, which is useful for instance when generating data from crowdsourcing. While different active learning strategies might work for the framework, the only strategy used in the work is active learning triggered by uncertainty.

The effects of an imperfect oracle are explored by Donmez et al., both with regard to not always being correct and to not always being avaliable [4]. To obtain labels, an active learning strategy based on uncertainty is used in the experiments. Unlike to the typical active learning setting, the oracle can be both fallible (i.e. provide incorrect labels) and reluctant (i.e. might not always respond when queried). The scenario discussed in the work is for batch learning however, and not streaming data.

Zeni et al. perform experiments where students are asked to provide information regarding their behaviour via a mobile application [16]. The information provided from the students is compared to information gathered from the phone, e.g. location, to test the correctness. The experiments show that the students do sometimes provide incorrect labels and that there was a variation among the individuals in the amount of incorrect labels provided.

Machine teaching where the user providing labels can have varying degree of reliability is an area that needs further investigation, as well as for single-pass streaming data in cold-start settings. In our work we take a step towards exploring how the reliability of a user affects performance of interactive online machine learning, including both machine teaching and active learning strategies.

3 Experimental setup

The aim of the experiments is to explore how varying reliability of a user providing labels affects performance of different online machine learning methods and different interactive learning strategies. The online learning setting means that the data arrives in a streaming fashion, where each instance is presented and processed once by the learning algorithm. This can be compared to pool-based settings where all unlabelled data instances typically are available at any given time.

In the experiments we have a cold start scenario, which means that there is no labelled data for the machine learning to train on at the start of the data stream. Labelled data has to be collected gradually over time to incrementally train the machine learning model.

Streaming data means that the total amount of data is not necessarily known and might even be infinite. This creates issues that has to be taken into account for the interactive learning strategies and the machine learning algorithms. For instance, the labelled data that is gradually collected cannot be stored indefinitely, as the amount of labelled data theoretically could approach infinity. To counteract this, there is a limit of how many labelled data instances are stored for each class. If the maximum limit is reached for a given class and a new instance with the same label arrives, the oldest one from the collection is discarded.

The evaluation was done in a test-then-train fashion [5], i.e. where the model first attempts to estimate the incoming data instance and then, if a label is provided, uses the new labelled instance to incrementally train the model. The result thus becomes an accumulative accuracy that showcases performance of the model over time. The results displayed in the next section are all average values of several runs of each experiments. How a run of the experiments was constructed is described in more detail in section 3.4.

3.1 Machine Learning algorithms

Three machine learning algorithms were implemented in the experiments, Naïve Bayes classifier, Support Vector Machine (SVM) and k-Nearest Neighbor (k-NN). The aim was to study if there is a discernible difference in the effect of a fallible or reluctant user. The algorithms were chosen to be well-known off-the-shelf machine learning algorithms but also suitable for the setting at hand, i.e. online learning with a cold start scenario. Even though the experiments presented here are from simulations done on previously recorded or created datasets, the intention is that the experiment should be able to run in real-time, which means that the complexity of the machine learning algorithm also has to be considered.

Gaussian Naïve Bayes classifier was included in the experiments because it works well for online learning and has low computational complexity [7]. It also needs a relatively small amount of training data before it can start to produce estimations, which makes it suitable for a cold start scenario.

SVMs are effective in dealing with high-dimensional data, which is of importance in many settings with streaming data, and have efficient memory usage [9]. SMVs aim to find hyperplanes that divides the classes with a margin that is maximized. As the number of fallible instances increases this will become increasingly difficult and the method will spend more time trying to optimize the classifier. To counteract that training time becomes a concern, a maximum number of iterations to optimize the hyperplanes is set to 1000. A polynomial kernel is used in the experiments, based on initial testing. These results are not included in this work due to space restrictions.

k-NN is a fitting classifier for our scenario because of its simplicity, it is suitable for online learning and the computational work needed can be limited through the amount of data temporarily stored [6]. The method looks at the labels of the k nearest, previously collected, data instances to classify a new data instance. This means that only k instances needs to be collected before the method can be used, which is good for a cold start scenario. The value of k was set to 3, based on initial experiments that are not included in this work due to space restrictions.

3.2 Interactive Learning strategies

Two different interactive learning strategies were used in the experiments, one active learning strategy, where the learning model is deciding when to query the user for a label, and one machine teaching strategy, where the user decides which instances to provide labels for. All type of interactive learning has to accommodate for a labelling budget. The labelling budget is set beforehand and decides how big portion of the total amount of data can be labelled by the user. In a pool-based setting, the use of the labelling budget is straightforward. In contrast, in a setting with streaming data, only one data instance is processed at a time and the choice of whether or not to ask for a label has to be done as soon as it appears in the data stream. Since the total number of data instances is unknown at the start of processing the streaming data, the labelling budget can not be calculated the same way as it is done in a pool-based setting. Instead, a sliding window containing information on which of the latest processed instances the user has provided a label for is used to calculate the current labelling expenses. The labelling expenses are compared to the labelling budget, to determine if it is currently possible to query for more labels. The window size is set to 200 instances in the experiments.

Active learning triggered by uncertainty Along with each estimation produced by the machine learning algorithm is also a measurement of how certain the model is of its own estimation. The active learning strategy compares the produced uncertainty measurement to a set threshold. If the measurement is below the threshold, the estimation is considered uncertain and the user is queried, given that there is enough labelling budget. This is the most common type of active learning used and is sometimes referred to as Uncertainty sampling [13] or Uncertainty-based sampling methods [8]. To implement this strategy, the measurement of uncertainty needs to be defined and this is dependent on the machine learning algorithm used. As three different machine learning algorithms are employed in our experiments, each one needs their own uncertainty measurement.

The Naïve Bayes classifier produces a probability for each class and then picks the class with the highest probability for its estimation. The probability of the chosen class is then compared to a threshold. An initial value is set for the threshold, but the value can be lowered or increased over time, depending on whether a query is made or not. If many queries are made the threshold is gradually lowered, as it might indicate that the threshold is too high, or the opposite if few queries are made. The implementation is based on the Variable Uncertainty Strategy presented by Žliobaitė et al. [19].

The measurement of uncertainty used for SVM is the distance from the new data instance to the hyperplanes. If the distance is short it means that the new instance is in close proximity to another class and thus might have a higher probability belonging to the other class compared to an instance further away. The distance is compared to a threshold that, like in the case of Naïve Bayes can be altered depending on how many queries are made.

For the k-NN the measurement is based on how many of the k instances nearest to the new instance, i.e. the instances deciding which class to estimate, have the same label. If more than two-thirds of the instances have the same label the estimation is considered certain otherwise not. The strategies used for SVM and k-NN are further described by Pohl et al. [11].

Machine teaching triggered by error In machine teaching it is the user that employs a strategy of when to provide labels to the learning algorithm. In the strategy included in the experiments, the user is aware of the current estimation produced by the model and whenever this estimation is incorrect the user is triggered to provide a label, given that there is enough labelling budget to do so. In this way the user can aid the model by correcting it when it makes a mistake.

3.3 Simulation of reliability in the user

In the experiments the aim was to explore how a varying degree of reliability in the user providing labels affects performance. The two aspects of reliability studied in the experiments were reluctance and fallibility, both are further explained below. By simulating reluctance and fallibility in the user, the level of reliability could be controlled in each experiment.

Reluctance A reluctant user is a user that does not always provide a label in accordance with the given learning strategy. In the case of the active learning strategy, the user will not always reply to a query and in the case of the machine teaching strategy, the user will sometimes not provide a label, even though the estimation is incorrect and there is enough labelling budget. In a real-world scenario the reluctant behavior can be explained by a user that i.e. is distracted by something else, unwilling to provide labels or uncertain of which label to provide. The level of reluctance is varied between 0% and 50% in the experiments, during which fallibility is kept at 0%. The level of reluctance informs how big portion of the queries posed that the user will not respond to. For each new query posed, a random number between 0 and 1 is generated and if the generated number is lower than the level of reluctance, the user will not reply.

Since the window used to calculate the current labelling expenses only includes the cases when a label has been received, the expenses are not increased when a query does not get a reply or when a user does not provide a label.

Theoretically this could mean that for the very next instance the active learning strategy could pose a new query or the user could provide a label in the case of machine teaching. This would not be very realistic however, as a user that for instance is distracted at one point in time will likely still be distracted the moment afterwards. Instead, if a label that should have been provided was not due to reluctance, the algorithm had to wait the length of the labelling window before another data instance could be considered for labelling.

Fallibility A fallible user does always provide a label when queried or triggered, but the label is not always correct. This could for example correspond to a user that does not know or is uncertain about the correct label or that makes a mistake. The experiments are constructed in a similar way to the experiments explained above for a reluctant user. The level of fallibility decides the probability of a label being incorrect. In the experiments the level is varied between 0% and 50%, while the level of reluctance was kept at 0%. When an incorrect label is provided, the false label to be attached to the data instance is chosen randomly from all the incorrect labels. As one of the datasets employed in the experiments contains real-world recordings (the mHealth dataset, described further below) there is a risk concerning the correctness of the labels provided. In the experiments however, the assumption is made that the labels in the dataset are correct.

3.4 Datasets

To study the effects on performance of reluctance and fallibility in the user, experiments were performed on two separate datasets. The first is an activity recognition dataset and consists of recordings from a real-world scenario. The second is a synthetically constructed dataset.

mHealth dataset The mHealth dataset consists of recordings of 10 subjects with wearable sensors performing a specific routine of physical exercises [1, 2]. The set of wearable sensors include gyroscope magnetometer, accelerometer and electrocardiogram sensor. Each recording contains between 98304 to 161280 data instances from one subject, resulting in 10 recordings in total. The data contained unlabelled data which was excluded for the experiments, resulting in recordings of a length 32205-35532 instances. The specific routine consists of 12 different physical exercises that the subject is meant to perform in a specific sequence. The routine is constructed so that one exercise follows the other, but is never repeated. Because of the test-then-train evaluation, the different classes that are to be estimated, i.e. the psychical exercises in this dataset, should appear more than once to result in any proper conclusion with regards to performance evaluation. To create a sequence where all exercises appear more than once, the recordings are all put after one another to create one longer data sequence. The order of which the different recordings are placed is randomly generated for each run. The result produced is the average of 20 separate runs.

Synthetic dataset The synthetically generated dataset[1] contains 50000 data instances in total. The dataset has two features and five classes with 10000 instances belonging to each class. For each class, a mean value was created for the two features. The instances was then generated by sampling from a 2D normal distribution with the given mean value of the given class and a set standard deviation. In Fig. 1 a visualisation of the dataset is displayed. For each run of the experiments, an order of all the data instances had to be established. This was done by first randomly choosing one of the classes, then a random sample from a normal distribution was drawn to decide how many samples of this class should be in the interval. If for instance 20 instances of class A was set for one interval, 20 random data instances belonging to class A were chosen and put after one another in the sequence. One interval was put after another until all the data instances were arranged in the sequence. The ordering of the instances was redone for each run, but the data instances themselves were the same for all of them. The result presented is the mean value of 100 separate runs.

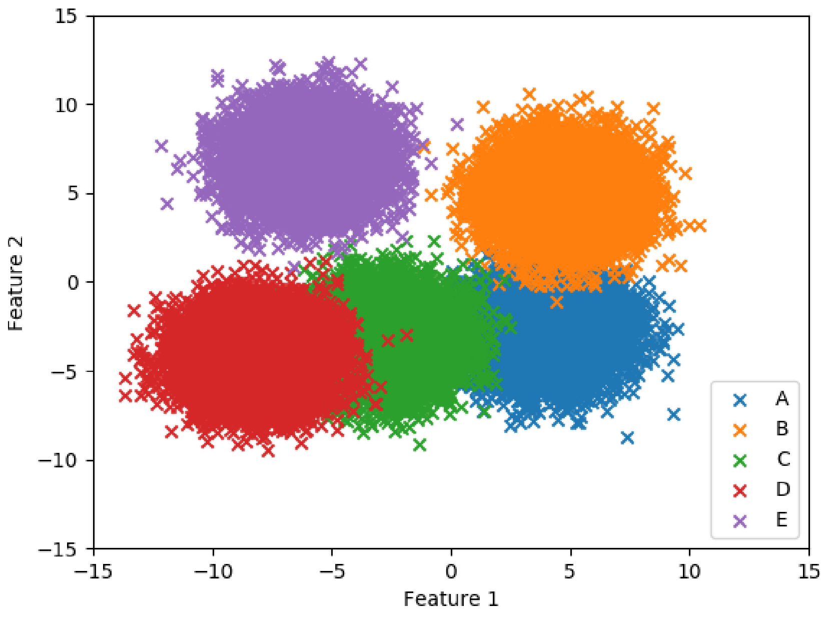


Fig. 1: A visualisation of the distributions of the classes of the synthetic dataset.

[1] The dataset can be found via the link: https://github.com/ategen/synthetic-dataset

3.5 Limitations

There are several aspects of the typical assumptions made in interactive learning that does not always hold in real-world scenarios. In this work we have chosen to focus on two aspects of reliability of the user, reluctance and fallibility, but there are other that could be of interest to study, depending on the application.

In the experiments it is assumed that there is one user, or multiple users with the same levels of reliability, providing labels. In certain settings however, there might be multiple users with different characteristics, all with the possibility to provide labels at least at some point in time. An example where this is highly relevant is in cases when crowdsourcing is used to collect labelled data.

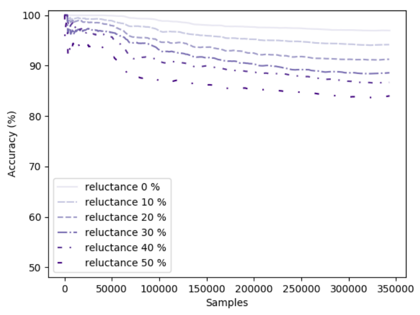
Another assumption made in the experiments that could be challenged is that the cost of providing a label is uniform for all possible labels. In some settings this might not be the case. For instance, if a scenario has classes that are similar, there might be data instances that need a more thorough examination to find out which class they belong to, while other instances can be classified by the user at a glance. If the cost of labelling varies, this could also connect to reluctance and fallibility of the user. A more difficult, or costly, label can lead to a higher probability that the user does not provide a label. There could also be a case where there is an option to have more thorough and costly labelling by the user, resulting in a lower fallibility, or a quicker and cheaper labelling, but with a higher risk of being fallible. Depending on the application setting, one approach might be preferable over the other.

One relevant issue when discussing learning from streaming data is concept drift, which means that the statistical properties of the streaming data changes over time. It is a phenomenon that is present in many streaming data settings and there exists works that discusses it in more detail [5]. Concept drift is not the main focus of this work, but is passively handled by continuously updating the machine learning model with new incoming data instances and discarding old ones. With a limited amount of data for training however, there is a risk of overfitting. When dealing with streaming data, this is an important trade-off to be aware of.

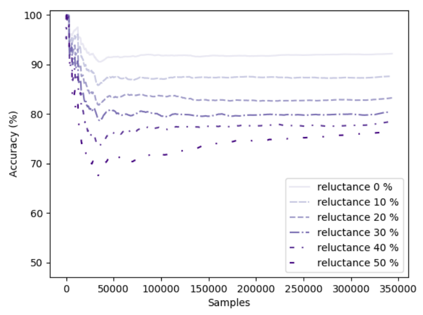
4 Results and Discussion

In Figs. 2 and 3 the results from experiments on the mHealth dataset when varying the degree of reluctance and fallibility of the user can be seen. Figs. 4 and 5 show results from the corresponding experiments on the synthetic dataset. The performance is displayed as accumulated accuracy over the number of samples that have been estimated so far. This means that when the number of samples is lower it represents performance early, i.e. when fewer estimations have been done and fewer labelled data instances have been gathered.

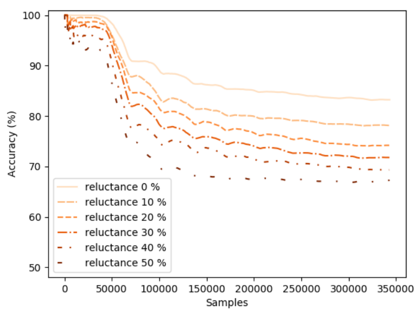
The results confirms that performance gets worse with an increased level of reluctance or fallibility of the user in all of the experiments. How big the decrease in performance is depends on the dataset, interactive learning strategy and machine learning method however.



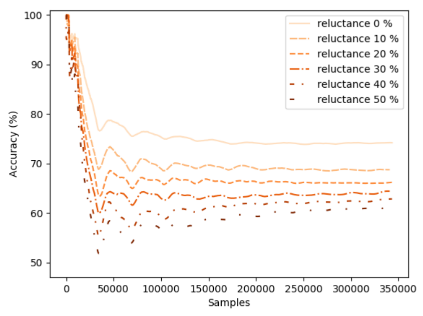
(a) Machine teaching, NB



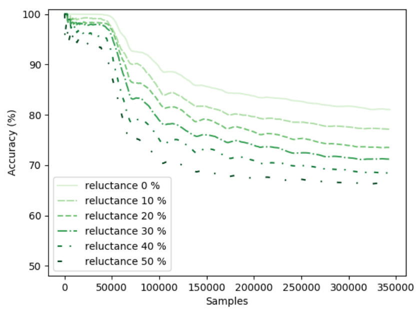
(b) Active learning, NB



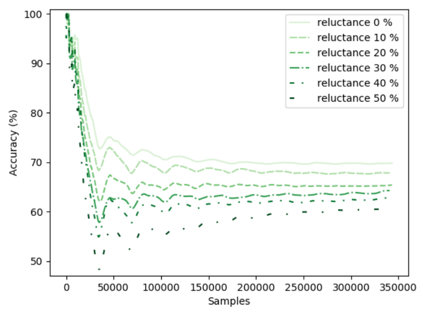
(c) Machine teaching, SVM



(d) Active learning, SVM

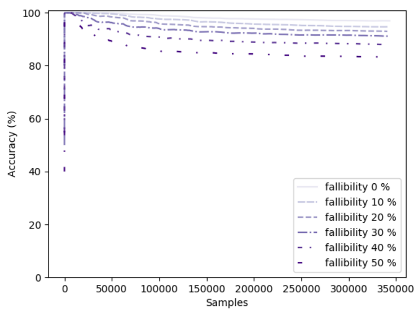


(e) Machine teaching, kNN

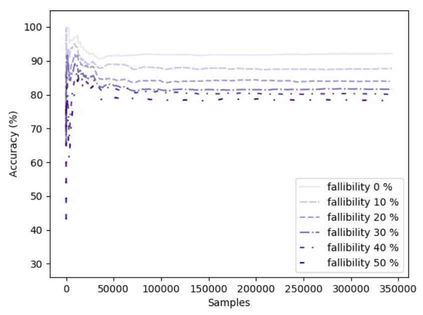


(f) Active learning, kNN

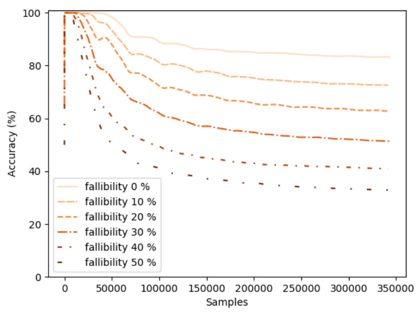
Fig. 2: The results from the experiments on the mHealth dataset when the level of reluctance is varied. The left column (a, c and e) displays the result for the machine teaching strategy triggered by uncertainty and the right column (b, d and f) for the active learning strategy triggered by error for Naïve Bayes (NB), Support Vector Machine (SVM) and k-Nearest Neighbor (kNN) respectively.



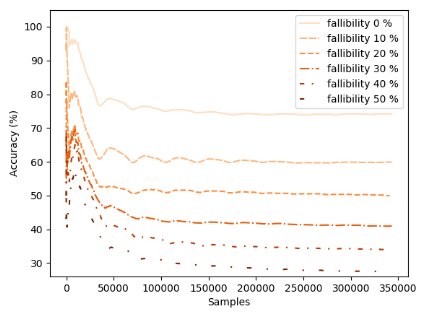
(a) Machine teaching, NB



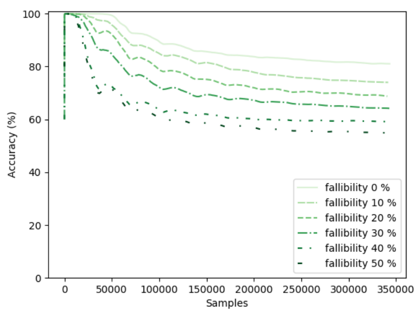
(b) Active learning, NB



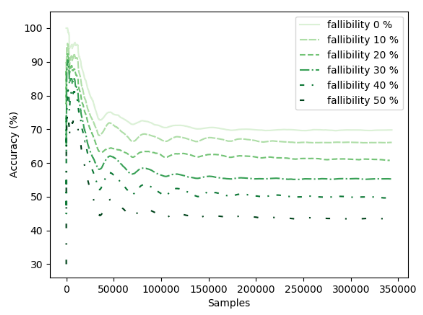
(c) Machine teaching, SVM



(d) Active learning, SVM

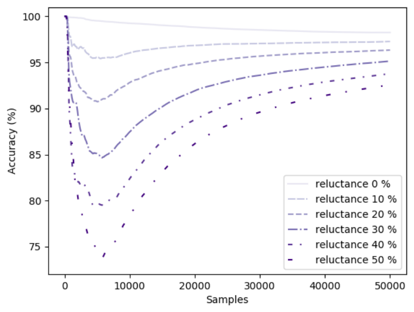


(e) Machine teaching, kNN

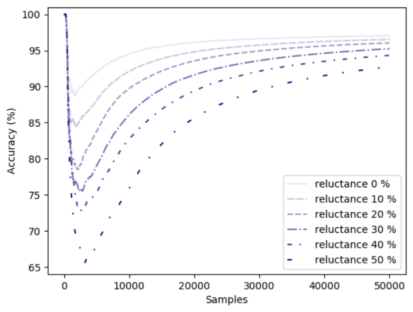


(f) Active learning, kNN

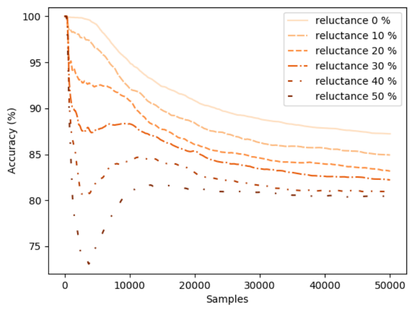
Fig. 3: The results from the experiments on the mHealth dataset when the level of fallibility is varied. The left column (a, c and e) displays the result for the machine teaching strategy triggered by uncertainty and the right column (b, d and f) for the active learning strategy triggered by error for Naïve Bayes (NB), Support Vector Machine (SVM) and k-Nearest Neighbor (kNN) respectively.



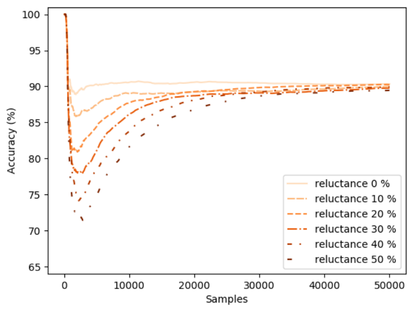
(a) Machine teaching, NB



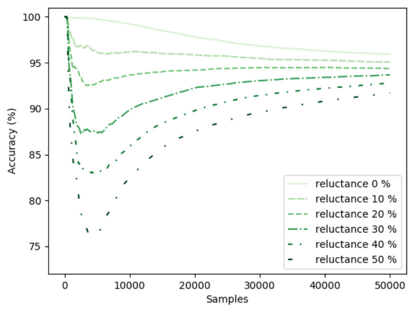
(b) Active learning, NB



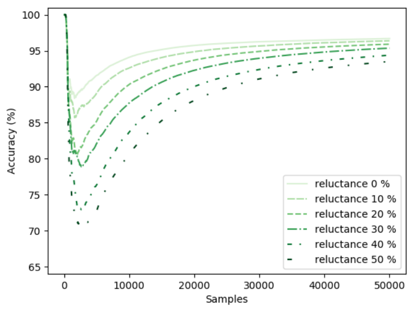
(c) Machine teaching, SVM



(d) Active learning, SVM

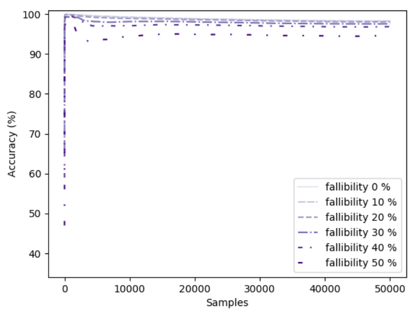


(e) Machine teaching, kNN

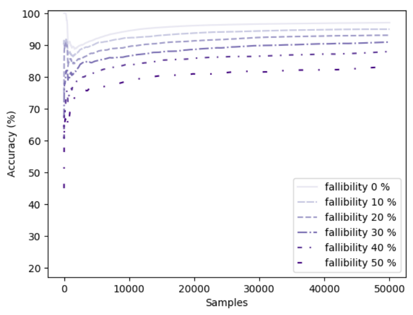


(f) Active learning, kNN

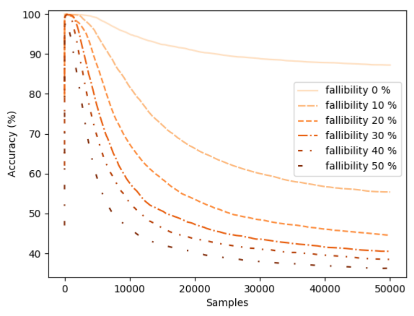
Fig. 4: The results from the experiments on the synthetic dataset when the level of reluctance is varied. The left column (a, c and e) displays the result for the machine teaching strategy triggered by uncertainty and the right column (b, d and f) for the active learning strategy triggered by error for Naïve Bayes (NB), Support Vector Machine (SVM) and k-Nearest Neighbor (kNN) respectively.



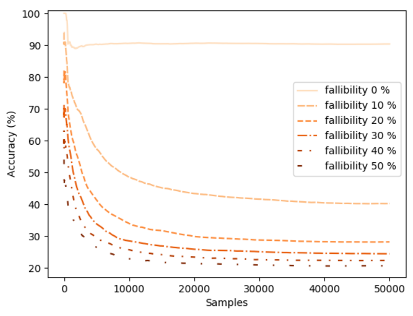
(a) Machine teaching, NB



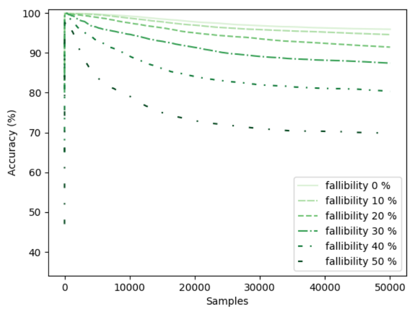
(b) Active learning, NB



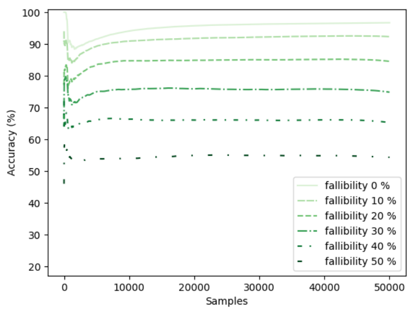
(c) Machine teaching, SVM



(d) Active learning, SVM



(e) Machine teaching, kNN



(f) Active learning, kNN

Fig. 5: The results from the experiments on the synthetic dataset when the level of fallibility is varied. The left column (a, c and e) displays the result for the machine teaching strategy triggered by uncertainty and the right column (b, d and f) for the active learning strategy triggered by error for Naïve Bayes (NB), Support Vector Machine (SVM) and k-Nearest Neighbor (kNN) respectively.

In the experiments on the mHealth dataset, Naïve Bayes classifier and the machine teaching strategy triggered by error has the highest performance. If the level of reluctance is increased (Fig. 2) or if the level of fallibility is increased (Fig. 3) the combination of Naïve Bayes and the machine teaching strategy is still giving the best performance. In Fig. 2a, showing the results for this combination with an increasing reluctance level however, the trend still seems to be downwards towards the end of all samples. This can be compared to Naïve Bayes with active learning triggered by uncertainty, Fig. 2b, where the performance overall is lower, but stabilizes after a while. In the experiments with a varying degree of fallibility, SVM and k-NN appear to be more affected compared to Naïve Bayes classifier. As the level of fallibility increases, the performance of SVM and k-NN drops faster, especially compared to Naïve Bayes in combination with the active learning strategy.

Figs. 4 and 5 contain the results from the experiments on the synthetic dataset. In the experiments where the reluctance of the user is varied, Fig. 4, the biggest difference in performance can be found early, when the number of samples is low. The main reason for this is likely that a higher level of reluctance in the user leads to a longer time before enough labelled data instances have been gathered to result in a performance on the same level as a user with 0% reluctance to provide labels. Towards the end of the average run, the performance of the different levels of reluctance approaches each other. When looking at the final accumulated accuracy of Fig. 4, Naïve Bayes classifier and k-NN performs better than SVM. The figure also shows that at the start the machine teaching strategy performs better than the active learning strategy. This is consistent over all machine learning algorithms tested, but the difference decreases as the level of reluctance increases. For SVM at the highest level of reluctance, there is no significant difference at the start between the active learning strategy and the machine teaching strategy. Furthermore, after a while the active learning strategy outperforms the machine teaching strategy.

In the results a drop in performance can be seen in several of the figures. One reason for this is the cold start scenario and the patterns in the data streamed. At the very start of the data stream, a labelled data instance from the first class is provided to the machine learning algorithm. In accordance with the nature of the data, for a period, the same class will continue and more labelled instances from this class can be collected. At this point in time the task of classifying is easier, or even trivial in the case of no fallibility. As more classes are introduced over time however, the difficulty of classification increases, which in turn can lead to a lower performance.

The experiments where the effects of fallibility of the user was tested on the synthetic dataset are displayed in Fig. 5. Here, the effect that the choice of interactive learning strategy and machine learning method can have on performance is clear. The best performing and most robust combination is Naïve Bayes classifier with the machine teaching strategy triggered by error. When the level of fallibility is at 0%, there is not a significant difference between the different interactive learning strategies and machine learning algorithms. When the fallibility level is higher than 0% however, the difference becomes noticeable. The biggest decrease in performance can be seen in the experiments using SVM, displayed in Figs. 5c and 5d. Since SVM tries to optimize the positioning of hyperplanes to as much as possible separate the different classes, the task will get increasingly more difficult as the number of incorrect labelled instances increases in the dataset used for training. The steepest drop in performance of these two can be seen in Fig. 5d, where the active learning strategy triggered by uncertainty is employed. The measurement of uncertainty implemented for SVM is based on the distance from the new data instance to be tested, to the hyperplanes. If the SVM classifier has trouble positioning the hyperplanes correctly due to incorrect training data, the uncertainty measurement which is dependent on this position will also be inadequate. While less extreme, the effect of an increasingly larger portion of training data being incorrect is visible for the Naïve Bayes classifier and k-Nearest Neighbor as well.

An interesting observation from the experiments is that the Naïve Bayes classifier appears more robust towards fallibility compared to reluctance while the opposite is true for SVM and k-Nearest Neighbor. The possible explanation for the poor results of SVM when fallibility is introduced is discussed above. For k-Nearest Neighbor, the higher the level of fallibility, the bigger the risk that the k closest instances are incorrect, which in turn leads to a faulty classification. For SVM and k-Nearest Neighbor the experiments show that it is better with a user that might not provide as many labels, but when they do they are correct. Naïve Bayes classifier on the other hand is a generative model which classifies by used mean values generated from the labelled data obtained. Depending on the nature of data, the averaging can smooth possible noise in the data and still create useful mean values. For Naïve Bayes classifier, the experiments indicate that a user who provides more data, even though some instances have incorrect labels, is preferable to a user that is more restrictive but always correct.

The experiments with a fallible user are meant to simulate a user that is not always correct in assessing what label currently is representative of the state to be classified. In a real-world scenario, a user that is sometimes incorrect in this assessment might not always recognize when the estimation of the machine learning algorithm is incorrect either. This is not included in the experiments where the machine teaching strategy is employed and might therefore not portray the entire spectrum of possible effects of a fallible user.

Another factor of the experimental setup that might affect the results is the choice of which data instances that are affected by fallibility or reluctance. As explained in section 3.3, the data instances that are either not provided, in the case of a reluctant user, or provided with incorrect labels, in the case of a fallible user, are chosen at random. In certain scenarios it might be reasonable to assume that the probability of all the instances to be chosen are evenly distributed. For instance, if the user is distracted by another task they are performing, they might sometimes, i.e. in a random pattern, miss to provide a label in accordance with the given learning strategy. For a user that is attentive but not as knowledgeable of what the correct label is on the other hand, the probability of which labels are not provided or given an incorrect label might be correlated to the data instance itself. For example, a data instance belonging to one label, but that is close to the boundary of another, might be more difficult for the user than a data instance that is a typical example of the same class.

5 Conclusion and Future Work

In this work we explored how the reliability of the user providing labels affects the performance of online machine learning in a cold start scenario. We also studied the robustness of different interactive learning strategies and different machine learning algorithms with regards to a user that can be fallible and reluctant respectively. The results show that the choice of interactive learning strategy and machine learning algorithm has an effect on performance in the experiments, where the combination of Naïve Bayes classifier and the machine teaching strategy triggered by error overall resulted the highest performance. This combination is also most robust towards increased levels of fallibility and reluctance of the user. The overall least robust machine learning algorithm was SVM, especially for a fallible user.

In future work we plan to further validate our conclusions by testing on other datasets and more machine learning algorithms. We also aim to further explore how varying the level of reliability of a user can affect performance.

References

1. Banos, O., Garcia, R., Holgado-Terriza, J.A., Damas, M., Pomares, H., Rojas, I., Saez, A., Villalonga, C.: mhealthdroid: a novel framework for agile development of mobile health applications. In: International workshop on ambient assisted living. pp. 91–98. Springer (2014)
2. Banos, O., Villalonga, C., Garcia, R., Saez, A., Damas, M., Holgado-Terriza, J.A., Lee, S., Pomares, H., Rojas, I.: Design, implementation and validation of a novel open framework for agile development of mobile health applications. Biomedical engineering online **14**(2), S6 (2015)
3. Bouguelia, M.R., Nowaczyk, S., Santosh, K., Verikas, A.: Agreeing to disagree: active learning with noisy labels without crowdsourcing. International Journal of Machine Learning and Cybernetics **9**(8), 1307–1319 (2018)
4. Donmez, P., Carbonell, J.G.: Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In: Proceedings of the 17th ACM conference on Information and knowledge management. pp. 619–628. ACM (2008)
5. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM computing surveys (CSUR) **46**(4), 1–37 (2014)
6. Khan, Z.A., Samad, A.: A study of machine learning in wireless sensor network. Int. J. Comput. Netw. Appl **4**, 105–112 (2017)
7. Krawczyk, B.: Active and adaptive ensemble learning for online activity recognition from data streams. Knowledge-Based Systems **138**, 69–78 (2017)
8. Lughofer, E.: On-line active learning: a new paradigm to improve practical useability of data stream modeling methods. Information Sciences **415**, 356–376 (2017)

9. Mahdavinejad, M.S., Rezvan, M., Barekatain, M., Adibi, P., Barnaghi, P., Sheth, A.P.: Machine learning for internet of things data analysis: A survey. Digital Communications and Networks **4**(3), 161–175 (2018)
10. Miu, T., Missier, P., Plötz, T.: Bootstrapping personalised human activity recognition models using online active learning. In: 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing. pp. 1138–1147. IEEE (2015)
11. Pohl, D., Bouchachia, A., Hellwagner, H.: Batch-based active learning: Application to social media data for crisis management. Expert Systems with Applications **93**, 232–244 (2018)
12. Ramdas, A., Poczos, B., Singh, A., Wasserman, L.: An analysis of active learning with uniform feature noise. In: Artificial Intelligence and Statistics. pp. 805–813 (2014)
13. Settles, B.: Active learning literature survey. Tech. rep., University of Wisconsin-Madison Department of Computer Sciences (2009)
14. Shickel, B., Rashidi, P.: Art: an availability-aware active learning framework for data streams. In: The Twenty-Ninth International Flairs Conference (2016)
15. Yan, S., Chaudhuri, K., Javidi, T.: Active learning from imperfect labelers. In: Advances in Neural Information Processing Systems. pp. 2128–2136 (2016)
16. Zeni, M., Zhang, W., Bignotti, E., Passerini, A., Giunchiglia, F.: Fixing mislabeling by human annotators leveraging conflict resolution and prior knowledge. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **3**(1), 1–23 (2019)
17. Zhu, X.: Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In: Twenty-Ninth AAAI Conference on Artificial Intelligence (2015)
18. Zhu, X., Singla, A., Zilles, S., Rafferty, A.N.: An overview of machine teaching. arXiv preprint arXiv:1801.05927 (2018)
19. Žliobaitė, I., Bifet, A., Pfahringer, B., Holmes, G.: Active learning with drifting streaming data. IEEE transactions on neural networks and learning systems **25**(1), 27–39 (2013)

Active Learning for LSTM-autoencoder-based Anomaly Detection in Electrocardiogram Readings

Tomáš Šabata[1] and Martin Holeňa[2]

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

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

Keywords: Active Learning, Anomaly detection, LSTM-Autoencoder, Time series

1 Introduction

Recently, the amount of generated time series data has been increasing rapidly in many areas such as healthcare, security, meteorology and others. However, it is very rare that those time series are annotated. For this reason, unsupervised machine learning techniques such as anomaly detection are often used with such data. There exist many unsupervised algorithms for anomaly detection ranging from simple statistical techniques such as moving average or ARIMA till complex deep learning algorithms such as LSTM-autoencoder. For a nice overview of the recent algorithms we refer to read [2,1].

Difficulties with the unsupervised approach are: defining an anomaly score to correctly represent how anomalous is the time series, and setting a threshold for that score to distinguish between normal and anomaly data. Supervised anomaly detection, on the other hand, needs an expensive involvement of a human expert. An additional problem with supervised anomaly detection is usually the occurrence of very low ratio of anomalies, yielding highly imbalanced data.

In this extended abstract, we propose an active learning extension for an anomaly detector based on a LSTM-autoencoder. It performs active learning using various classification algorithms and addresses data imbalance with over-sampling and under-sampling techniques. We are currently testing it on the ECG5000 dataset from the UCR time series classification archive [3].

2 Active learning for LSTM-autonecoder-based anomaly detection

LSTM-autoencoder [9] is nowadays increasingly used to detect anomalies in time series data [11,5,4]. The algorithm aims to learn the identity function. It consists of two parts – an encoder and a decoder. The encoder compresses the input representation of the data into a low-dimensional latent representation (usually called *code*) from which the decoder reconstructs the original input. The model parameters are found by minimizing the reconstruction error. Samples are then considered anomalous if their reconstruction error is higher than a selected threshold.

Although LSTM-autoencoder works well for time-series with complicated patterns, setting the anomaly score threshold can be very complicated without labelled data. Furthermore, with a higher ratio of anomalies present in the training data, a simple setting of the threshold might produce a lot of false positives and false negatives. Therefore, we incorporated active learning into building the anomaly detector that uses the code of a previously trained LSTM autoencoder. The most related research has been done by Pigmentel [8], who proposed to use a classifier (logistic regression) with the latent layer of autoencoder together with anomaly score. In contrast, we experiment with the latent layer of a recurrent autoencoder without anomaly score and propose to use resampling techniques.

First, an LSTM autoencoder is trained on unlabelled data. An initial anomaly detection threshold on reconstruction error distribution is selected. At the initial value of the threshold, we decided to use the mean plus three times the standard deviation of the reconstruction error as an initial threshold. Every sample with the reconstruction error above the threshold is labelled as an anomaly and the same number of samples below the threshold are labelled as normal. Furthermore, instead of the original time-series data, we use their representation in the code layer. This provides us an artificially created, balanced dataset and converts the anomaly detection to a binary classification task. A classifier is trained on the created dataset and an active learning loop starts. In each iteration, a resampler is used to balance the new updated labelled dataset. The resampler can be either an undersampling or oversampling algorithm. The classifier is fitted using the resampled data. Uncertainty sampling (US) active learning framework [7] is used to select instances which should be labelled by an oracle. We use margin US, i.e. we select instances leading to the smallest difference between the likelihood of anomalous and normal data classes. Anomaly detection is then based on predictions of that classifier instead of on the anomaly threshold. The pseudo-code for the algorithm is shown in Algorithm 1.

3 Experiment and Results

The proposed algorithm was evaluated on a benchmark time-series dataset with electrocardiogram readings [3]. Each heart-beat record in the dataset is labelled with one of 5 classes where the last three classes are very rare and we consider them as an anomaly. The dataset was split into training, validation and testing in the ratio 70:15:15. The validation dataset was used to find the hyperparameters for the LSTM-autoencoder. The autoencoder achieving the lowest f1 score in the anomaly detection on the validation data set was chosen. The final architecture of the encoder consisted of two LSTM cells. The cells have one hidden layer with 48 neurons in the first cell and 24 neurons in the second cell. The hidden state

Algorithm 1: Active learning for LSTM-autoencoder anomaly detection

```
Input:
$\mathcal{U}$: unlabelled data set of sequences
$\theta$: anomaly score threshold
$\phi(\cdot)$: query strategy utility function
begin
    train LSTM autoencoder on data set $\mathcal{U}$
    // calculate anomaly scores
    $a_i = |x_i - \text{decoder}(\text{encoder}(x_i))|, x \in \mathcal{U}$
    sort $a$ in the descending order
    sort $x$ respectively to $a$
    $i = 0, \mathcal{L} = \emptyset$
    // Add anomalous data samples into the labelled dataset
    while $a_i > \theta$ do
        $\mathcal{L} = \mathcal{L} \cup \langle \text{encoder}(x_i), 1 \rangle$
        $\mathcal{U} = \mathcal{U} \setminus x_i$
        $i = i + 1$
    end
    // Add normal data samples into the labelled dataset
    for $j = i$ to $2i$ do
        $\mathcal{L} = \mathcal{L} \cup \langle \text{encoder}(x_j), 0 \rangle$
        $\mathcal{U} = \mathcal{U} \setminus x_j$
    end
    while stopping criterion is not met do
        $\mathcal{R} = \text{resampler}(\mathcal{L})$
        train binary classifier $m$ on $\mathcal{R}$
        // Find the most informative seqeuence from $\mathcal{U}$ and ask for label
        $x^* = \text{argmax}_{x \in \mathcal{U}} \phi(x)$
        $y^* = \text{query}(x^*)$
        $\mathcal{L} = \mathcal{L} \cup \langle x^*, y^* \rangle$
        $\mathcal{U} = \mathcal{U} \setminus x^*$
    end
end
```

of the last cell is copied and used as the input of the first cell of the decoder. The decoder has architecture mirrored.

We experimented with 5 classifiers: logistic regression, decision tree, Gaussian naive Bayes classifier, k-nearest neighbours classifier and support vector machines. We experimented with 11 under-sampling and 4 over-sampling techniques taken from the imbalanced-learn python toolbox [6]. In the presented results, we report 5 under-sampling techniques that were best on average. In the experiment, we compared a fully unsupervised approach, in which anomalies are detected by using a chosen threshold, with our extension with respect to F1 score.

Figure 1 shows how actively asking for annotations can improve the unsupervised anomaly detection with an LSTM-autoencoder (red dashed line). The best results were achieved with repeated edited nearest neighbours [10] and a k-nearest neighbour classifier. However, using an SVM as the base classifier yielded more stable performance.

Moreover, a classifier model fed by labels outperformed the LSTM-autoencoder with the initially set anomaly score threshold (red dashed line) and the best possible anomaly score threshold (green dashed line). The source code is available in GitHub repository [3].

4 Conclusion

We presented an active Learning for LSTM-autoencoder-based anomaly detection for time-series data. An experiment on the ECG5000 data set has shown that the proposed method is able to boost the performance of the model significantly with only approximately 200 labelled samples. We plan next to experiment with variational LSTM-autoencoders and to pay attention to the interpretability of the detected anomalies .

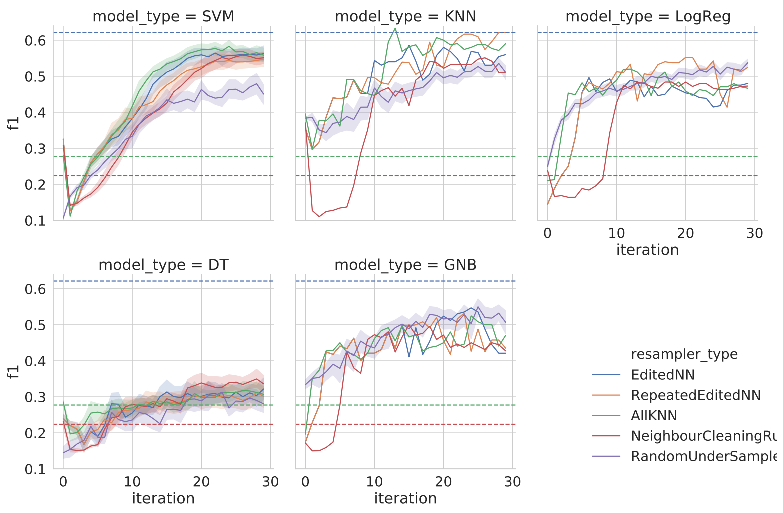
Acknowledgements

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

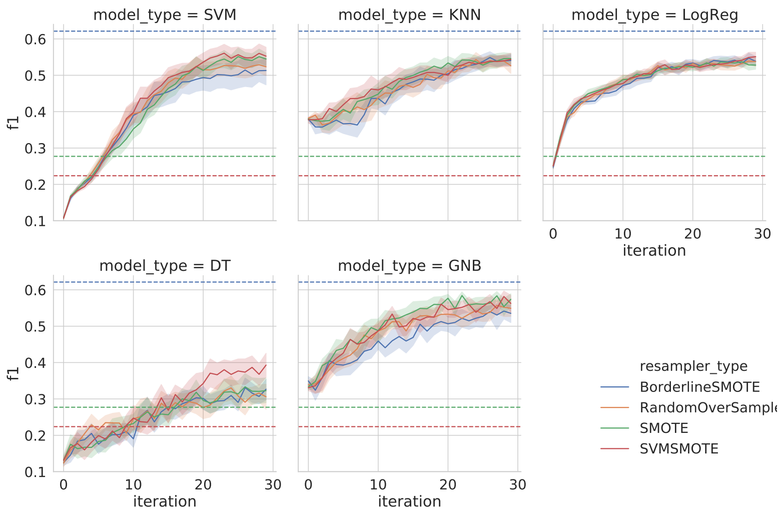
References

1. Ahmad, S., Lavin, A., Purdy, S., Agha, Z.: Unsupervised real-time anomaly detection for streaming data. Neurocomputing **262**, 134–147 (2017)
2. Cook, A., Mısırlı, G., Fan, Z.: Anomaly detection for iot time-series data: A survey. IEEE Internet of Things Journal (2019)
3. Dau, H.A., Keogh, E., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G., Hexagon-ML: The ucr time series classification archive (October 2018), `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`
4. Ergen, T., Mirza, A.H., Kozat, S.S.: Unsupervised and semi-supervised anomaly detection with lstm neural networks. arXiv preprint arXiv:1710.09207 (2017)
5. Guo, Y., Liao, W., Wang, Q., Yu, L., Ji, T., Li, P.: Multidimensional time series anomaly detection: A gru-based gaussian mixture variational autoencoder approach. In: Asian Conference on Machine Learning. pp. 97–112 (2018)
6. Lemaître, G., Nogueira, F., Aridas, C.K.: Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. The Journal of Machine Learning Research **18**(1), 559–563 (2017)
7. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: SIGIR'94. pp. 3–12. Springer (1994)
8. Pimentel, T., Monteiro, M., Viana, J., Veloso, A., Ziviani, N.: A generalized active learning approach for unsupervised anomaly detection. stat **1050**, 23 (2018)

[3] `https://github.com/tsabata/active_anomaly_detection`



(a) Under-sampling techniques.



(b) Over-sampling techniques.

Fig. 1: F1 score performance metric in active learning loop. The figure contains 5 models (support vector machines (SVM), k-nearest neighbours (KNN), logistic regression(LogReg), decision tree (DT) and Gaussian naive Bayes(GNB)) and 9 resampling techniques. The greyed area represents standard deviation. Red dashed line represents the performance of the LSTM-autoencoder anomaly detector without active learning and initial setting of the anomaly score threshold. Green dashed line represent the performance of the LSTM-autoencoder anomaly detector without active learning and the best setting of the anomaly score threshold, and blue dashed line represents the best-achieved performance in the last iteration of active learning loop.

9. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: International conference on machine learning. pp. 843–852 (2015)
10. Tomek, I., et al.: An experiment with the edited nearest-nieghbor rule. IEEE Transactions on Systems, Man, and Cyberbetics **6**, 448–452 (1976)
11. Zhang, C., Chen, Y.: Time series anomaly detection with variational autoencoders. arXiv preprint arXiv:1907.01702 (2019)

Towards Landscape Analysis in Adaptive Learning of Surrogate Models

Zbyněk Pitra[1,2] 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, martin}@cs.cas.cz
[2] Faculty of Nuclear Sciences and Physical Engineering, CTU in Prague
Břehová 7, 115 19 Prague 1, Czech Republic

Keywords: Adaptive learning · Optimization strategy · Black-box optimization · Landscape analysis · Surrogate model

1 Introduction

A context in which we expect adaptive learning to be promising is the choice of a suitable optimization strategy in black-box optimization. The reason why strategy adaptation is needed in such a situation is that knowledge of the black-box objective function is obtained only gradually during the optimization. That knowledge covers two aspects:

1. the landscape of the black-box objective, revealed through its evaluation in previous iterations;
2. success or failure of the optimization strategies applied to that black-box objective in previous iterations.

To extract landscape knowledge, landscape analysis has been developed during the last decade [7,10,11]. To include also the second aspect, we complement features obtained using the landscape analysis with features describing the optimization employed in previous iterations.

Our interest is in expensive black-box optimization, where the number of evaluations of the expensive objective is usually decreased using a suitable surrogate model. Therefore, the research reported in this extended abstract addresses adaptive learning of surrogate models, more precisely their learning in surrogate-assisted versions of the state-of-the-art black-box optimization method, Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [3].

Considering the results in [2,14] suggesting that the properties of landscape features in connection with surrogate model selection problem should be analysed in more detail, we contribute with this work a first essential step towards a better understanding, by analysing the robustness of feature computation. Such analysis of a large set of landscape features has already been presented only in connection with selection of the most convenient optimization algorithm for problems in fixed dimension [15].

This extended abstract focuses on surrogate model selection task in multiple dimensions and discusses robustness of several classes of features against samples of points from the same distribution.

2 Landscape Analysis for Surrogate Model Selection

Landscape analysis aims at measuring characteristics of the objective function using functions that assign to each dataset a set of real numbers [10]. Let's consider a dataset of N pairs of observations $\left\{(\mathbf{x}_i, y_i) \in \mathbb{R}^D \times \mathbb{R} \cup \{\circ\} \mid i = 1, \ldots, N\right\}$, where $\circ$ denotes missing y_i value (e. g., $\mathbf{x}_i$ was not evaluated yet). Then the dataset can be utilized to describe landscape properties using a feature $\varphi : \bigcup_{N \in \mathbb{N}} \mathbb{R}^{N,D} \times (\mathbb{R} \cup \{\circ\})^{N,1} \mapsto \mathbb{R} \cup \{\pm\infty, \bullet\}$, where $\bullet$ denotes impossibility of feature computation.

Feature classes convenient for continuous black-box optimization field are mostly described in [7]. From the available feature classes we mention only those convenient for problems with a high computational complexity (unlike e. g., cell-mapping approach [8]) and at the same time not requiring additional evaluations of the expensive function. Feature classes are able to measure the dissimilarity among points of a subset of the sample (*Dispersion*) [9], express various information content of the landscape (*Information Content*) [11], measure the relative position of each value with respect to quantiles (*Levelset*) [10], extract the information from linear or quadratic regression models (*Meta-Model*) [10] or from the nearest or the better observation neighbours (*Nearest Better Clustering*) [6], and describe the distribution of the objective values (*y-Distribution*) [10]. Moreover, in [13] we have proposed the set of features based on the CMA-ES state variables (*CMA features*).

The surrogate model selection problem tackle the situation in an iteration i of a surrogate-assisted algorithm A, where a set of surrogate models $\mathcal{M}$ are trained using a training set $\mathcal{T}$ selected out of an *archive* $\mathcal{A}$ ($\mathcal{T} \subset \mathcal{A}$) of all points evaluated so far using the objective function f: $\mathcal{A} = \{(\mathbf{x}_i, f(\mathbf{x}_i)) \mid i = 1, \ldots, N\}$. Hereafter, a new set of points $\mathcal{P} = \{\mathbf{x}_k | k = 1, \ldots, \alpha\}$ is evaluated using a surrogate model $M \in \mathcal{M}$, where $\alpha \in \mathbb{N}$ depends on the strategy defining the usage of surrogate model in algorithm A. The research question is: How to select the most convenient M from $\mathcal{M}$ according to $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{P}$?

To tackle the research question connected with the surrogate model selection problem, we have proposed (see [14]) the following metalearning approach visualised in Figure 1:

In the first phase, each model $M \in \mathcal{M}$ is trained on each $\mathcal{T}^{(l)}$ from the set of datasets $\mathcal{D} = \{\mathcal{A}^{(l)}, \mathcal{T}^{(l)}, \mathcal{P}^{(l)}\}_{l=1}^{L}$, $L \in \mathbb{N}$ and its error ε is measured on $\mathcal{P}^{(l)}$. Simultaneously, a set of features Φ is computed on each dataset from $\mathcal{D}$. Hereby, a mapping $S_M : \Phi \to \mathcal{M}$ from the space of landscape features to $\mathcal{M}$ is trained. In the second phase, the trained mapping S_M is utilized in each iteration i of the algorithm A to select the model $M \in \mathcal{M}$ according to the features Φ calculated on $\mathcal{A}^{(i)}$, $\mathcal{T}^{(i)}$, and $\mathcal{P}^{(i)}$. The selected M is utilized to fit $\mathcal{T}^{(i)}$ and afterwards to predict objective function values of points from $\mathcal{P}^{(i)}$.

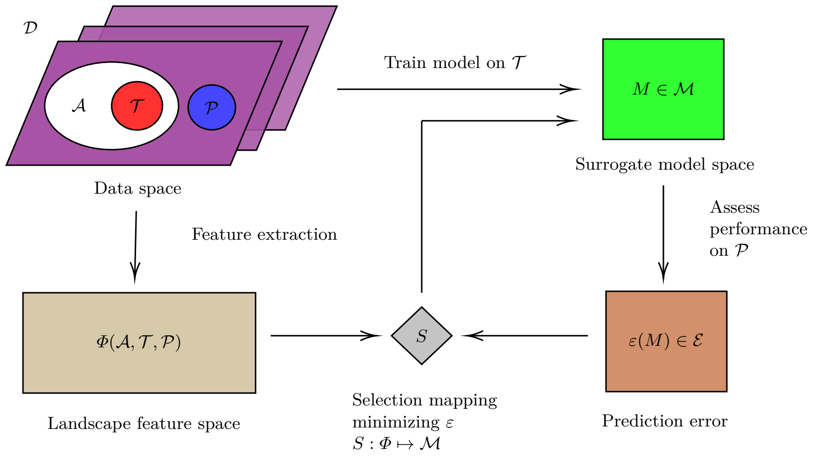

$\mathcal{D}$
$\mathcal{A}$
$\mathcal{T}$
$\mathcal{P}$
Data space
Train model on $\mathcal{T}$
$M \in \mathcal{M}$
Surrogate model space
Assess performance on $\mathcal{P}$
Feature extraction
$\Phi(\mathcal{A}, \mathcal{T}, \mathcal{P})$
Landscape feature space
S
Selection mapping minimizing ε $S : \Phi \mapsto \mathcal{M}$
$\varepsilon(M) \in \mathcal{E}$
Prediction error

Figure 1: Scheme of the metalearning approach to the surrogate model selection system [14].

3 Feature Robustness

To investigate robustness of feature computation against different samples of points (in the sense of low variance), several independent archive realisations using the same distributions should be available. To gain such realisations, we have created a new set of artificial distributions by smoothing the distributions from real runs of the surrogate algorithm on the set of benchmarks.

First, we have generated a set of datasets $\mathcal{D}$ using independent runs of the 8 model settings from [13] for the DTS-CMA-ES algorithm [1,12] on the 24 noiseless single-objective benchmark functions from the COCO framework [4,5]. All runs were performed in dimensions 2, 3, 5, 10, and 20 on instances 11–15. To gain 100 comparable archives using those runs, we have generated points for new archives using the weighted sum of original archive distributions from $\mathcal{D}$, where the weight vector $\mathbf{w}^{(i)} = \frac{1}{9}(0, \ldots, \underset{i-3}{0}, \underset{i-2}{1}, \underset{i-1}{2}, \underset{i}{3}, \underset{i+1}{2}, \underset{i+2}{1}, \underset{i+3}{0}, \ldots, 0)^\top$ provides distribution smoothing across the available iterations[1]. Second, for all $\mathcal{A}^{(i)}$, $\mathcal{T}^{(i)}$, and $\mathcal{P}^{(i)}$ from $\mathcal{D}$ we have computed all features from the following feature classes: *Dispersion*, *Information Content*, *Levelset*, *Meta-Model*, *Nearest Better Clustering*, *y-Distribution*, *CMA features*.

Once the features are computed, the numbers of $\pm\infty$ and $\bullet$ values of different samples from one iteration are summarized and the rest of feature values is normalized to $[0, 1]$ range using feature minima and maxima over the whole $\mathcal{D}$. We then compare feature means and variances for individual iterations.

[1] Weighted sum of the original archive distributions satisfies $\sum_{n=0}^{i_{\max}} w_n^{(i)} \mathcal{N}\big(\mathbf{m}^{(n)}, \mathbf{C}^{(n)}\big) \sim \mathcal{N}\big(\sum_{n=0}^{i_{\max}} w_n^{(i)} \mathbf{m}^{(n)}, \sum_{n=0}^{i_{\max}} (w_n^{(i)})^2 \mathbf{C}^{(n)}\big)$, where $i_{\max}$ is the maximal iteration reached by particular original archive and $\mathbf{m}^{(n)}$ and $\mathbf{C}^{(n)}$ are mean and covariance matrix in iteration n.

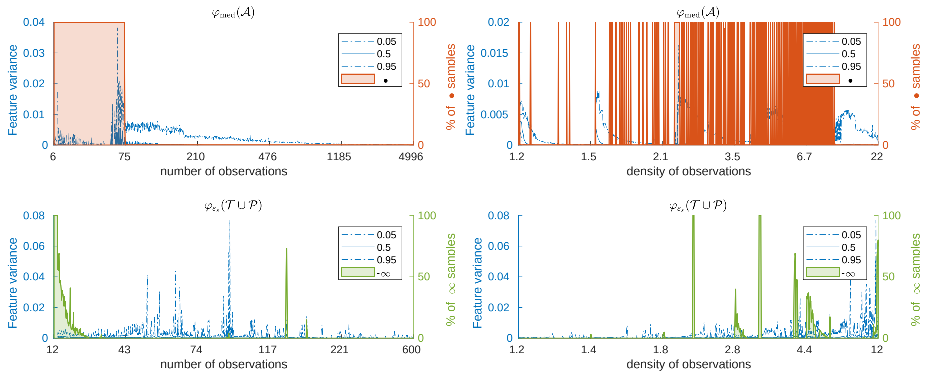


Figure 2: The dependecies of 0.05, 0.5, and 0.95 quantile of feature variance, the median number of $\pm\infty$, or $\bullet$ of feature values on the number of observations N for two features are shown on plots in the first column. The dependencies of the same statistics on the data density $\sqrt[D]{N}$ are presented in the second column. Plots in the first row represent statistics for feature $\varphi_{\mathrm{med}}(\mathcal{A})$ – median distance of the 'best' vs. 'all' objectives in $\mathcal{A}$ (from *Dispersion* feature class) and the second row contains statistics for $\varphi_{\varepsilon_s}(\mathcal{T} \cup \mathcal{P})$ – settling sensitivity of the information content in $\mathcal{T} \cup \mathcal{P}$ (*Information Content*).

Figure 2 shows the dependecies of 0.05, 0.5, and 0.95 quantile of feature variance, the number of $\pm\infty$, or $\bullet$ on the number of observations N in the considered set ($\mathcal{A}$, $\mathcal{T}$, or $\mathcal{P}$) and data density $\sqrt[D]{N}$ for two example features.

The results show that most of the features are robust in the sense of having a low variance, especially for higher numbers of observations. Robustness for lower values of N is not frequently high, or even the feature is not possible to calculate (e. g., some of *Dispersion* features). *CMA* features provided the most robust results probably due to the fact that most of them are sample independent. The lowest variance values, and also high numbers of cases where the feature was impossible to calculate were observed at *Dispersion* features.

4 Conclusion

The extended abstract addressed adaptive learning of a suitable optimization setting in black-box optimization, more precisely, adaptive learning of a surrogate model in a surrogate-assisted version of the CMA-ES. Its main message is the relationship of this kind of adaptive learning to landscape analysis. A formal framework for the learning of a surrogate model based on landscape analysis is given, and considered kinds of landscape features are discussed. In the results obtained so far, attention is paid in particular to feature robustness.

This work in progress is part of a thorough investigation of the possibilities of landscape analysis in the context of surrogate modelling for black-box optimization. That investigation has already brought first results in the past [2,13,14], but much still remains for further research.

Acknowledgements The reported research was supported by the Czech Science Foundation grant No. 18-18080S and by the Grant Agency of the Czech Technical University in Prague with its grant No. SGS20/183/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. Bajer, L., Pitra, Z., Repický, J., Holeňa, M.: Gaussian process surrogate models for the CMA Evolution Strategy. Evolutionary Computation **27**(4), 665–697 (2019)
2. Dvořák, M.: Optimization of surrogate model settings using landscape analysis. Master's thesis, Czech Technical University in Prague, Faculty of Information Technology, Prague, Czech Republic (2020)
3. 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)
4. Hansen, N., Auger, A., Finck, S., Ros, R.: Real-parameter black-box optimization benchmarking 2012: Experimental setup. Tech. rep., INRIA (2012)
5. 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
6. Kerschke, P., Preuss, M., Wessing, S., Trautmann, H.: Detecting funnel structures by means of exploratory landscape analysis. pp. 265–272. GECCO '15, ACM (2015)
7. Kerschke, P., Hoos, H.H., Neumann, F., Trautmann, H.: Automated algorithm selection: Survey and perspectives. Evolutionary computation **27**(1), 3–45 (2019)
8. 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)
9. Lunacek, M., Whitley, D.: The dispersion metric and the cma evolution strategy. pp. 477–484. GECCO '06, ACM (2006)
10. Mersmann, O., Bischl, B., Trautmann, H., Preuss, M., Weihs, C., Rudolph, G.: Exploratory landscape analysis. pp. 829–836. GECCO '11, ACM (2011)
11. 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)
12. 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)

13. 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)
14. Pitra, Z., Bajer, L., Holeňa, M.: Knowledge-based selection of Gaussian process surrogates. In: Kottke, D., Lemaire, V., Calma, A., Krempl, G., Holzinger, A. (eds.) ECML PKDD 2019: Workshop & Tutorial on Interactive Adaptive Learning. Proceedings. pp. 48–63. ECML PKDD 2019, Würzburg, Germany (Sep 2019)
15. Renau, Q., Dreo, J., Doerr, C., Doerr, B.: Expressiveness and robustness of landscape features. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. p. 2048–2051. GECCO '19, Association for Computing Machinery, New York, NY, USA (2019)

VIAL-AD: Visual Interactive Labelling for Anomaly Detection – An approach and open research questions

Andreas Theissler[1] [0000–0003–0746–0424], Anna-Lena Kraft[2], Max Rudeck[1], Fabian Erlenbusch[1]

[1] Aalen University of Applied Sciences, 73430 Aalen, Germany
[2] IT-Designers Gruppe, 73730 Esslingen, Germany

Abstract. In anomaly detection problems the available data is often not or not fully labelled. This leads to results that are usually significantly worse than in balanced classification problems. In this short paper VIAL-AD is proposed, which addresses this problem with a sequence of unsupervised, semi-supervised and supervised machine learning models allowing a user to interactively label data points. This allows to move towards supervised anomaly detection, starting with unlabelled data. The approach is introduced and identified open research questions are discussed.

Keywords: visual interactive labelling · VIAL · anomaly detection · human-centered machine learning

1 Introduction

This work addresses machine learning-based anomaly detection (AD) [6, 1], where the aim is to classify data points as either *normal* or *anomaly* based on a set of features f. This can be achieved by training AD models on data that is (a) unlabelled, (b) contains labelled normal data, or (c) contains labelled normal data and anomalies. Applications of AD can be found in system health monitoring, intrusion detection, fraud detection, and the analysis of medical data. One main application field is data-driven fault detection, e.g. addressed in [20, 21].

This work is motivated by the question of *how we can compensate for the lack of a labelled and representative data set in AD problems by incorporating human knowledge in order to move to supervised AD.* While there are statistical or unsupervised ML methods to identify outliers, only a human expert can decide whether a data point is a true anomaly for a given application. Therefore, it suggests itself to incorporate the user in the process. We argue that for anomaly detection, this is indeed even more crucial than for balanced classification problems.

In this paper *visual interactive labelling for anomaly detection (VIAL-AD)* is proposed which – starting with unlabelled data – allows to iteratively move from unsupervised to supervised anomaly detection [6]. This is achieved by a combination of (1) a sequence of machine learning (ML) models with different levels of supervision and (2) the incorporation of the user to interactively label data. The idea is to use unsupervised AD to address the so-called cold-start problem [22] in order to obtain an initial set of tentative labels. A sequence of AD models is used to suggest labels and a human expert confirms or overrules suggestions and labels data or regions in the feature space. We believe that in AD, where it is unlikely to have a representative and labelled training set, the user-in-the-loop is key to allow for the use of ML and move towards an accuracy that allows for productive use. In order to validate the idea, a prototype was implemented. Preliminary results are promising, however a number of open research questions were uncovered and are discussed in Section 3.

In the following, related work is briefly reviewed. Holzinger et al. showed how a user in-the-loop with ML models can improve the overall performance of a system [11]. A generic process for visual interactive labelling was proposed by Bernard et al. under the name of VIAL [4]. In [3] it was shown that VIAL can outperform pure active learning – specifically for two-class problems. In [2] AD models were used for interactive labelling, however not with the aim to label an AD data set. Trittenbach et al. discuss open research challenges for one-class active learning [22], e.g. the cold-start problem.

2 The approach: VIAL-AD

VIAL-AD consists of the steps *unsupervised*, *semi-supervised*, and *supervised AD* [6] (see Fig. 1(a)), where in each step model and user collaborate as follows: The model classifies the data and suggests the labels *normal* and *anomaly*. The user inspects the data points and their labels and (a) adjusts model hyperparameters, or (b) confirms/overrules the labels proposed by the model, or (c) visually labels data points or regions. Following that, the user decides to move to the follow-up step or to refine the labelling in the current step.

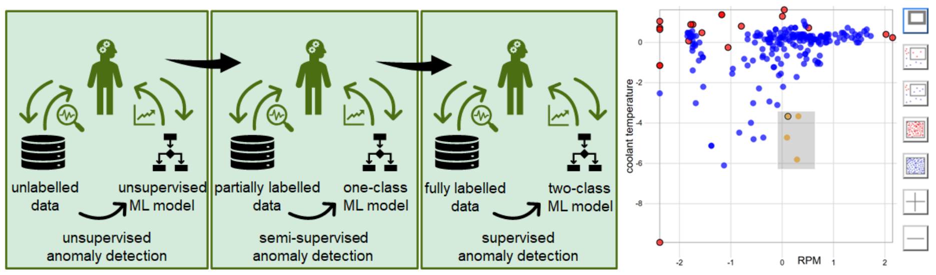


Fig. 1: (a) VIAL-AD: Interactive labelling with a sequence of ML models and the user-in-the-loop. (b) UI for visual interactive labelling, where the data shows RPM (revolutions per minute) and coolant temperature of vehicles during the occurrence of fault codes, read-out in car repair shops [19].

1. *unsupervised AD:* An unsupervised model (LOF [5] in current implementation) suggests initial labels which are confirmed or overruled by the user. Confirmed anomalies are moved to the two-class data set for the supervised step and are excluded for the current step. The model continues to report anomalies which are again evaluated by the user. In addition, regions in the feature space can be marked as *normal* or *anomaly* resulting in the creation of artificial data points. The result of the unsupervised step is a tentatively labelled train set.
2. *semi-supervised AD:* The reduced data set from the unsupervised step is used to train a one-class classifier (current prototype uses a OC-SVM [16]). For each data point the classifier suggests the labels *normal* or *anomaly*. These are processed analogously to the first step. A promising alternative is to make use of the anomalies from step 1 using methods like SVDDneg [18, 10]. This step's output is a labelled train set of normal data and anomalies, where the anomaly class is, however, not likely to be representative.
3. *supervised AD:* As VIAL-AD is used on real data, more and more anomalies are detected, so one can move to a supervised scenario. In addition to the normal data, the previously labelled anomalies are used to form a two-class train set that becomes increasingly representative. Hence, a variety of common ML models becomes applicable. In case of high class imbalance, sampling methods should be applied [12]. In the prototype ν-SVM is used.

The central element of interaction is an interactive 2D-scatter plot as in [2] (see Fig. 1(b)). This does however not limit the approach to two-dimensional data – higher dimensional data can be projected [15] onto two dimensions. Alternatively, visualisations for multi-dimensional data could be used. However, they introduce a higher complexity for the user.

3 Open research questions

Potential disruption caused by subsequent models: Different AD models have differing underlying assumptions [6, 18] about anomalies. Some work with probabilistic distributions, others with distances, densities (e.g. LOF [5]), reconstruction errors (autoencoders), or the adaption of the maximum-margin assumption to the one-class case (one-class SVMs [18, 16]). Hence, the use of different models in subsequent steps can induce disruptions in the way labels are suggested. A subsequent model could come up with a different labelling, which is confusing for the user. This disruption is to be minimised.

Visualisation-vs.-model dilemma: For data with > 2 dimensions the user is presented projected data, while the model may work on the original or on projected data. However, a projection with the aim to optimally visualise the data is not necessarily the optimal projection for the ML model to work on [14]. The original space or alternative projection methods might be more appropriate. Visualising and classifying different representations of the data can, however, induce undesired effects. Wenskovitch et al. give an overview and potential solutions are discussed in [24].

Problem of non-interpretability of projected space: In AD problems, typically no representative set of anomalies exists. To compensate for that, entire regions could be marked as anomalous based on expert knowledge. In the original feature space these would be outlier values that can be clearly specified by experts. As discussed, the potentially high-dimensional data can be projected onto a lower dimensional space, the user can interact with. However, for many projection methods the relation between the visualization and the original input space is not obvious. This makes it difficult or even impossible for the user to label unoccupied regions in the feature space. Projection methods like t-SNE [23] aim to preserve the neighbourhood between data points, however do not preserve the properties in unpopulated regions. This creates a dilemma trying to mark unpopulated regions as *normal* or *anomaly*: in contrast to working on the original feature space, users do not have an intuition about where anomaly regions in the projected space are, as the projected feature space can be distorted and hardly interpretable [15]. This problem can be addressed in several ways:

1. Avoid projections by using visualization methods for high-dimensional data, which however makes interaction with the data more complex and does not scale well for a high number of dimensions.
2. Show original data objects for selected data points. Data types, where single data objects can be intuitively presented due to some order within the data are predestined for that, e.g. images, time series, or text. High-dimensional data in the form of independent feature vectors can, however, not easily be represented in an intuitive way.
3. Investigate projection methods and interaction facilities in order to allow for a user-friendly interaction [9, 8]
4. Let the user explore different projections, e.g. as proposed in [7].

Problem of highly imbalanced data: In AD, the distribution of the classes is typically highly imbalanced towards the *normal* class. As a consequence (a) this poses particular challenges for the projection methods, and (b) it raises the question if users will label the data accordingly.

In projections, anomalies should be positioned well separated from normal instances. Hence, an interesting issue is the sensitivity of projection methods to outliers. Bernard et al. evaluated different projection methods in [2]. While PCA's sensitivity to outliers is considered promising [2], in [3] it is stated that users prefer t-SNE [23]. The appropriateness of a projection method can be evaluated with a user study or using metrics specifying the readability of the projections. In [17, 13] ways to measure this readability are discussed.

The second question raised by the class imbalance is if users will label the data accordingly: On the one hand, users might run into the risk of overlooking anomalies due to their rareness. On the other hand, it might be that users overestimate the *anomaly* class, labelling too many data points as anomalous.

Risk of manual overfitting: Furthermore, an identified challenge is the risk of what we call "manual overfitting". In supervised ML, overfitting is addressed e.g. with regularization terms – preventing the model from too naively overfitting the data. However, with the user in-the-loop and with direct control over class labels, such a naive overfitting may take place: The user might be tempted to process the data set in such a way to achieve optimal accuracy as opposed to strictly applying domain knowledge to distinguish between normal or anomalous data points or regions in the feature space. This could be addressed with a – potentially high number – of *blind* test sets. Even after testing, this data should not be made available to the expert in order not to overfit towards the test set. Ideally, a test set should only be used once to evaluate performance. Another option would be the introduction of some regularization method, putting reasonable constraints on the user actions.

4 Conclusion

This paper discussed how the incorporation of humans can compensate for the lack of labelled data in anomaly detection. The proposed approach uses unsupervised AD on an initially unlabelled data set and lets the user confirm or overrule decisions. After having interactively processed the data set in collaboration with unsupervised AD models, the user can move to semi-supervised or supervised models. The key benefit of VIAL-AD is, that it allows to move towards supervised ML where it was previously not applicable due to the lack of labelled data. This is a problem often encountered in industry where data is recorded for a different purpose and the opportunities of applying ML are discovered later. While preliminary experiments indicate the applicability of VIAL-AD, open research questions were identified which will be addressed in future work. Following that, the goal is to evaluate VIAL-AD in a systematic user study and to apply it in a real-world case study.

References

1. An experimental evaluation of novelty detection methods. Neurocomputing **135**, 313 – 327 (2014)
2. Bernard, J., Dobermann, E., Sedlmair, M., Fellner, D.W.: Combining cluster and outlier analysis with visual analytics
3. Bernard, J., Hutter, M., Zeppelzauer, M., Fellner, D., Sedlmair, M.: Comparing Visual-Interactive Labeling with Active Learning: An Experimental Study. IEEE Transactions on Visualization and Computer Graphics **24**(1), 298–308 (2018)
4. Bernard, J., Zeppelzauer, M., Sedlmair, M., Aigner, W.: VIAL: a unified process for visual interactive labeling. The Visual Computer **34**(9), 1189–1207 (2018)
5. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: Identifying Density-Based Local Outliers. In: SIGMOD Conference. pp. 93–104 (2000)
6. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Computing Surveys **41**(3), 15:1–15:58 (Jul 2009)

7. Cutura, R., Holzer, S., Aupetit, M., Sedlmair, M.: VisCoDeR: A tool for visually comparing dimensionality reduction algorithms. In: 26th European Symposium on Artificial Neural Networks, ESANN 2018, Bruges, Belgium, April 25-27, 2018 (2018)
8. Endert, A., Fiaux, P., North, C.: Semantic interaction for sensemaking: Inferring analytical reasoning for model steering. IEEE Transactions on Visualization and Computer Graphics **18**(12), 2879–2888 (2012)
9. Faust, R., Glickenstein, D., Scheidegger, C.: DimReader: Axis lines that explain non-linear projections. IEEE Transactions on Visualization and Computer Graphics **25**(1), 481–490 (2019)
10. Görnitz, N., Kloft, M., Rieck, K., Brefeld, U.: Toward supervised anomaly detection. J. Artif. Int. Res. **46**(1), 235–262 (Jan 2013)
11. Holzinger, A., Plass, M., Kickmeier-Rust, M., Holzinger, K., Crişan, G.C., Pintea, C.M., Palade, V.: Interactive machine learning: experimental evidence for the human in the algorithmic loop. Applied Intelligence **49**(7), 2401–2414 (2019)
12. Krawczyk, B.: Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence pp. 1–12 (2016)
13. Micallef, L., Palmas, G., Oulasvirta, A., Weinkauf, T.: Towards Perceptual Optimization of the Visual Design of Scatterplots. IEEE Transactions on Visualization and Computer Graphics **23**(6), 1588–1599 (2017)
14. Sacha, D., Sedlmair, M., Zhang, L., Lee, J.A., Peltonen, J., Weiskopf, D., North, S.C., Keim, D.A.: What you see is what you can change: Human-centered machine learning by interactive visualization. Neurocomputing **268**, 164–175 (2017)
15. 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. IEEE Transactions on Visualization and Computer Graphics **23**(1), 241–250 (2017)
16. Schölkopf, B., Platt, J.C., Shawe-Taylor, J.C., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Computation **13**, 1443–1471 (July 2001)
17. Sedlmair, M., Tatu, A., Munzner, T., Tory, M.: A taxonomy of visual cluster separation factors. Computer Graphics Forum **31**(3pt4), 1335–1344
18. Tax, D.M.: One-class classification. Concept-learning in the absence of counterexamples. Ph.D. thesis, Delft University of Technology (2001)
19. Theissler, A.: Multi-class novelty detection in diagnostic trouble codes from repair shops. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). pp. 1043–1049 (2017)
20. Theissler, A.: Detecting anomalies in multivariate time series from automotive systems. Ph.D. thesis, Brunel University London (2013)
21. Theissler, A.: Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection. Knowledge-Based Systems **123**(C), 163–173 (May 2017)
22. Trittenbach, H., Englhardt, A., Böhm, K.: Validating one-class active learning with user studies – A prototype and open challenges
23. van der Maaten, L., Hinton, G.: Visualizing Data using t-SNE. Journal of Machine Learning Research **9**(11), 2579–2605 (2008)
24. Wenskovitch, J., Crandell, I., Ramakrishnan, N., House, L., Leman, S., North, C.: Towards a systematic combination of dimension reduction and clustering in visual analytics. IEEE Transactions on Visualization and Computer Graphics **24**(1), 131–141 (2018)

Get a Human-In-The-Loop: Feature Engineering via Interactive Visualizations

Dimitra Gkorou[1], Maialen Larrañaga[2], Alexander Ypma[1], Faegheh Hasibi[3], Robert Jan van Wijk[1]

[1] ASML, Veldhoven, the Netherlands
{dimitra.gkorou,alexander.ypma,robert-jan.van.wijk}@asml.com
[2] Tessella, Toulouse, France
maialen.mlz@gmail.com
[3] Radboud University of Nijmegen, the Netherlands
f.hasibi@cs.ru.nl

Abstract. In manufacturing, data sets tend to be high-dimensional, with a low number of labels and, features show spurious correlations with respect to a target *key performance indicator*. As a consequence, costly manual feature engineering by domain experts is required prior to prediction. To improve this process, we propose an interactive feature engineering scheme based on dimensionality reduction. Low-dimensional embeddings of selected features are visualized and guide the domain experts towards effective feature engineering. We show that by engineering features we obtain higher predictive capabilities and we improve the interpretability of the model.

1 Introduction and Background

Many applications of predictive Machine Learning (ML) require significant Feature Engineering (FE) when having small datasets. Particularly in Integrated Circuit (IC) manufacturing, which is the application of this work, the absence of large amounts of labeled data, the requirements of interpretability and the already mature domain knowledge make FE crucial for predictive tasks. Recently, deep learning has shown potentiality in automatically generating useful features. However, the data requirements for obtaining good accuracy with deep learning i.e., about 5000 labeled instances for decent performance and about 10 million labeled instances for outstanding performance [1, Chap. 1], are not realistic for IC manufacturing. Also, most of our usecases require interpretability because ML predictions are expected to contribute to decisions on fab processes with high financial impact and so, highly complex models are not applicable. Finally in IC manufacturing, FE typically requires knowledge on the physics of a process which cannot be easily obtained by statistical techniques. As a result, FE is a costly process which is performed by domain experts manually.

We propose an interactive FE scheme, with a human expert in-the-loop, based on state-of-the-art dimensionality reduction. We implemented an initial approach of it as a web application in ASML, the leading manufacturer of lithography machines and major player in the semiconductor industry. Our scheme is an iterative process. In each iteration an expert observes an embedding of selected features and acquires some information based on the cluster structure present in it. For example, they understand which features are responsible for the clusters in the embedding or they understand what context the clusters belong to. After having observed the embedding, they then provide a set of rules (e.g., a clustering). This information is used to engineer a new feature. Expert input is obtained through visualizations. The human-defined clustering encodes (1) their prior knowledge on the underlying predictive task and, (2) the knowledge acquired through unexpected or surprising structure observed in the embedding. The previously observed patterns are factored out from the embedding. A new iteration begins with the expert observing the new embedding for clustering that can used for a new feature. The interaction ends when at a given iteration the embedding shows no more relevant clustering.

Related Work Our work has been motivated by the works in `Tiler` [3], and `SIDE` [7]. These pioneering works construct *informative* visualizations that are tailored for each particular user, based on their prior knowledge. However, these works consider linear dimensionality reduction (DR) which is not suitable for the complexity of our data sets. When using linear DR, we often see no cluster structure. [8] and [4] propose non linear DR for informative visualizations: conditional Variational Autoencoders and conditional t-SNE respectively. These methods can be used in our interactive scheme in order to construct embeddings that guide domain experts towards feature engineering.

Use case IC manufacturing is a complex process where various machines and processing tools are used such as coating, exposure or etching tools. ICs are being fabricated on a thin silicon plate, called *wafer*, which is processed in several *layers*. Sensors monitor each step of this process. The raw measurements of these sensors are the *features* used to predict *Key Performance Indicators* (KPIs). As an example of KPI, the work in [2] aims to predict the precision in *nanometers* of printing IC designs on a wafer, called *overlay*, using sensor measurements and context information. Overlay is measured over several positions on a wafer after each process layer. The features have a direct physical relationship with the KPI. The tools associated with the sensors are the *context* of the measurements. Unlike features, the context variables such as tool names, machine settings, time stamps, do not have a direct physical relationship with the KPIs. Nevertheless, context variables are necessary in order to explain the raw sensor measurements [5]. Predicting KPIs only based on the raw measurements gives low accuracy models. Typically, sensor measurements are noisy, with redundancies, and have offsets depending on their context namely, associated tools used in a particular step. Moreover, the tools are not always matched with respect to a common reference. For these reasons, enhancing these models with features properly engineered by domain experts is crucial. To evaluate our interactive scheme, we consider the prediction of a KPI in [2] which is a typical use case of our domain.Detailed description of the use case and the features can be found in [2].

2 Feature Engineering with Human in the Loop

In this section, we describe the proposed methodology for feature, how the interactive scheme is implemented and the improvements we obtain by engineering the features during the interaction.

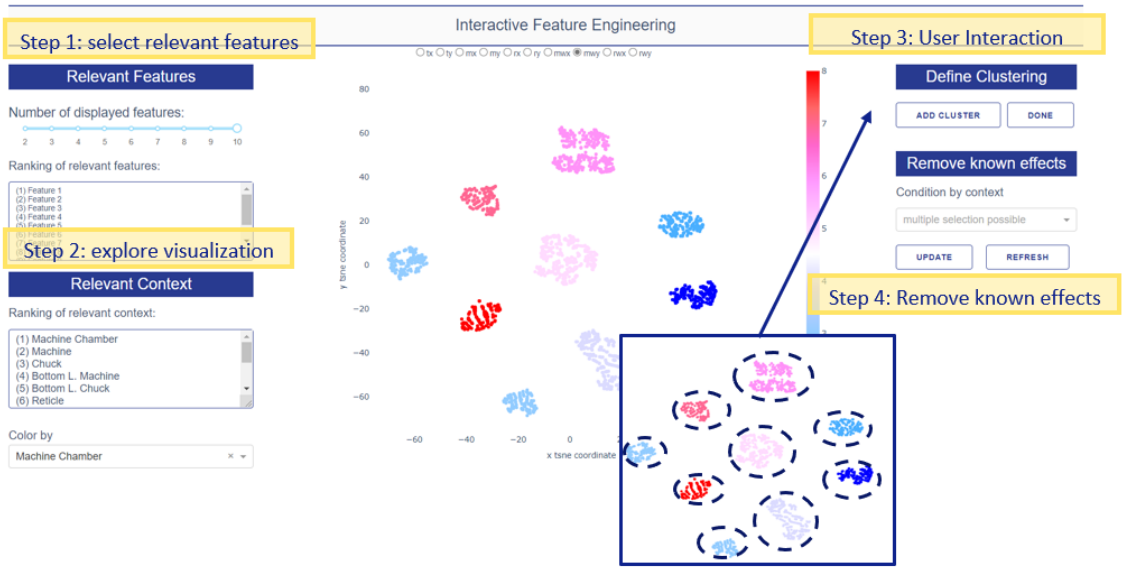


Fig. 1: Software design and steps to follow in the interactive scheme.

2.1 Methodology and SW tool

We developed a web application tool in `DASHPlotly` in python 3.7. It has four main steps as shown in Fig. 1. Below we describe it in detail.

Step 1: [*Select relevant features and visualize data*] Raw features together with context information is given as input to the software in a tabular format. Then an algorithm ranks the most relevant features with respect to the target KPI. Feature selection is out of the scope of this work because any feature selection could be used without affecting the FE interactions. For the feature selection, we use Bayesian Regression [6]. The user can select the number of ranked relevant features that will be used for the visualization, in Fig. 1 10 are picked. Based on those 10 features data is visualized in 2D using well-known dimensionality reduction methods. In Fig. 1, we used t-SNE as we have a small dataset. In a setting with large datasets, dimensionality reduction techniques such as UMAP [9] and Variational Autoencoders [8] can be used.

Step 2: [*Explore visualization*] On the left panel, the user can choose the context with which the scatter plot is colored. It can be the machine where the wafer has been exposed(which is selected in Fig. 1), the Reticle that has been used in the exposure, etc. The expert sees what context explains the clustering or structure in the data. To facilitate this, the context variables are ranked according to their Mutual Information with the clustering on the embedding as defined by Hierarchical Density-Based Spatial Clustering (HDBSCAN). So the user can start exploring the embedding using the context that correlates the most to its structure. As an example of the knowledge that the expert can acquire in this phase, we can see the *context effects* in Fig. 1 which can be explained by the machine chamber settings.

Step 3: [*User interaction for Feature Engineering*] The clustering observed in the previous step can be added as cluster constraints by the user. In this way, the user engineers a feature by computing the cluster-offset per machine setting,

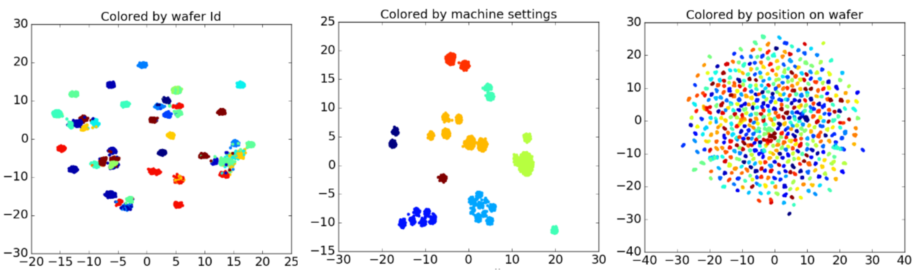


Fig. 2: Each embedding refers to an iteration of the interactive FE scheme.

namely the average of the target KPI per cluster. In fact, the new feature is a form of *target encoding* the machine context variable.

Step 4: [*Remove known effects*] The user continues the exploration in order to discover less dominant context-effects. To achieve this, we consider the previously acquired knowledge as prior information and we factor it out from the embedding using conditional t-SNE [4]. In the panel in Fig.1, the user is able to select the context-effect to be removed. Multiple selections are possible. The interaction starts again with a new visualization.

The interaction stops when the user cannot make sense out of the structure on the data anymore. In that step, we assume that every effect that could be *easily* understood from the data is known by the human expert and a feature has been engineered for it.

2.2 Application to the Overlay Prediction Use Case

We now present the iterations of human interactions with the proposed FE scheme for the KPI (overlay) prediction use case described in Section 1. First, we describe what a human expert learned by observing the embedding and what kind of features were engineered. Then, we evaluate the impact of the engineered features on overlay prediction.

We have a data set that consists of $\sim$ 2000 wafers. Overlay is measured in 60 points on each wafers and so, in total, our data has $120,000$ datapoints. Overlay is a continuous value and thus, we have a regression problem. After feature selection process, from an initial data set of $\sim$ 350 features, we have 30 features. To avoid overfitting, we used $\sim$ 10% of data for FE and the rest of it for training models. In Fig. 2 we see each iteration of the FE scheme.

Iteration 1: The data in the first plot of Fig. 2 is colored by wafer Id, which explains most of the clustering structure. This means that the overlay measurements on a wafer (independently of the layer they have been measured) look similar. In our data set, the measurements on a wafer are always taken with respect to the same reference layer. This means that if there is a distortion in the reference layer it will propagate through stack to all the layers above. Overlay experts can quickly identify this behavior. They now need to make sure that the initially loaded data contains the overlay measurements of at least one previous layer per wafer or the reference layer. In this case, two features can be engineered; the estimated cluster average and the overlay measurements of a previous layer.

Iteration 2: The structure observe in the $2nd$ embedding in Fig. 2 is obvious for an expert who can quickly relate it to overlay. It belongs to 8 different machine settings. Here the engineered feature is simply the estimated averages per setting.

Table 1: Prediction performance (r^2) for three iterations of the FE scheme.

	Bayesian Ridge	Random Forest
Baseline	0.0107	0.4537
1st iteration	0.3057	0.4636
2nd iteration	0.5863	0.601
3rd iteration	0.589	0.6027

Iteration 3: The last iteration shows a clustering that is colored per position on wafer. It is known that measurements of errors on the edge of a wafer are larger than those on the interior rings of the wafer. This structure again is easy for an expert to relate to overlay errors. The engineered feature is just the expected error on each position of the measurement.

In order to evaluate the accuracy of the model we implement a linear and a non-linear ML algorithm; Bayesian Ridge Regressor and Random Forest from scikit-learn. In Table 1, we evaluate two ML algorithms at the different phases of the interaction. The reported results derive from a 3-fold cross validation.

The first row, i.e., *baseline*, refers to the accuracy obtained by using the raw input data after feature selection. We see that the linear regressor was not able to capture the contributors to overlay because typically the signals are related to the overlay in a non-linear fashion. Random forest is able to capture quite some of the effect with the raw input data. In the next three rows, from *1st iteration* to *3rd iteration*, in each step new features have been added. We see that, the more features we engineer the better r^2 values we obtain. Another key message is that, once we have engineered all features, the results by a linear machine and a non-linear learning machine become comparable. In this example, the accuracy that we have obtained is not really high ($r^2 \sim 0.6$). Overlay prediction is a challenging task, and a result of 0.6 after a few FE steps is quite good.

Why do we need interpretable features? In this example, experts could easily identify the structures in each iteration. They were able to iteratively obtain all three *effects* contributing to the target KPI; distortions on reference layer, scanner settings and measurement position. Typically, context effects are dominant and can be easily identified by experts in data. However, some data sets might have more complex clustering structures that domain experts cannot relate with context variables. In IC manufacturing, a field heavily relying on the physics of the process, it is preferred to have an interpretable and less predictive feature than a more predictive but not well understood one. Experts usually discard predictive features that cannot explain as spurious correlations. In the proposed FE scheme, the interpretability of features by the experts is a requirement. At least some of the clusters of the embedding have to be explained by the context variables in order to engineer a feature.

Why do we need a human expert? One could argue that if the clustering in the visualization is obvious, a clustering algorithm such as DBSCAN could be run to do FE automatically. However, as we have seen in Iteration 1, the engineered features where not simple cluster average estimations, we also engineered another feature based on the knowledge an expert has on the domain. Also, clustering is an ill-posed problem in the sense that different clustering algorithms or different initializations of the same algorithm might give different results on the same data. Having an expert in the loop makes sure that not only the hidden structure in the data will be properly captured by engineered features, but also the domain knowledge will be used. The model becomes more interpretable, since a few human defined features are used as input data.

3 Conclusions and Open Questions

Our proposed FE scheme facilitates experts to engineer features in a structured way using their domain knowledge. The proposed scheme lends itself very naturally for semiconductor applications, but may be just as applicable in situations with high complexity, small sample sizes and existence of relevant (but maybe implicit) domain knowledge, e.g. medicine, general industry settings, etc. Nevertheless, we still face several challenges in implementing the proposed FE scheme:

First, we would like to *transfer* the learned features from one domain to another. Similar context effects are present in many predictive machine learning settings over different domains within semiconductor industry. The reason is that we typically want to predict KPIs from sensor measurements associated with some context. Our FE scheme requires costly time from domain experts. Transferring the learned features across domains, instead of requesting similar input from expert in different domains, will improve its efficiency.

Second, domain experts are often unsure of their feedback and they might also be biased. Making our scheme robust to biases and conflicting inputs is necessary for its successful adoption.

References

1. I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning.* MIT press, 2016.
2. F. Hasibi, L. van Dijk, M. Larrañaga, A. Pastol, A. Lam, and R. van Haren. Towards fab cycle time reduction by machine learning-based overlay metrology. In *34th European Mask and Lithography Conference*, pages 129 – 137. SPIE, 2018.
3. A. Henelius, E. Oikarinen, and K. Puolamäki. Tiler: Software for human-guided data exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 672–676. Springer, 2018.
4. B. Kang, D. García García, J. Lijffijt, R. Santos-Rodríguez, and T. De Bie. Conditional t-sne: Complementary t-sne embeddings through factoring out prior information, 2019.
5. A. Lam, A. Ypma, M. Gatefait, D. Deckers, A. Koopman, R. van Haren, and J Beltman. Pattern recognition and data mining techniques to identify factors in wafer processing and control determining overlay error. *Proc. SPIE*, 9424, 2015.
6. M Larranaga, D Gkorou, T Guzella, A Ypma, F Hasibi, and RJ van Wijk. Towards Interactive Feature Selection with Human-in-the-loop. In *Workshop on Interactive Adaptive Learning*, 2018.
7. J. Lijffijt, B. Kang, K. Puolamäki, and T. De Bie. Side: a web app for interactive visual data exploration with subjective feedback. In *ACM SIGKDD Workshop on Interactive Data Exploration and Analytics (IDEA)*, 2016.
8. A. Marot, A. Rosin, L. Crochepierre, B. Donnot, P. Pinson, and L. Boudjeloud-Assala. Interpreting atypical conditions in systems with deep conditional autoencoders: the case of electrical consumption. In *ECML PKDD*, 2019.
9. L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.