Exploring the Unknown – Query Synthesis in One-Class Active Learning

Adrian Englhardt[*] Klemens Böhm[*]

Abstract

The quality of a classifier hinges on the availability of training data. In scenarios where data collection is restricted or expensive, e.g., compute-intensive simulations, training data may be small and/or biased. In principle, data synthesis then allows to extend the data set. Yet it is difficult for a user to extend the data without any guidance when the data space is unbound or of high dimensionality. In this article we target at the *domain expansion* problem, i.e., expanding the classifier knowledge beyond an initial sample that completely falls into one class. We first propose a general framework for query synthesis in the one-class setting. Then we present a new query synthesis strategy to quickly explore the data space beyond the initial sample. For the evaluation we derive three options to simulate an oracle in the one-class setting that can answer arbitrary queries. Experiments on both synthetic and real world data demonstrate that our new query strategy indeed expands the knowledge of a one-class classifier beyond a small and biased initial sample. Our strategy outperforms realistic baselines on most domain expansion problems.

Keywords one-class classification, active learning, query synthesis, domain expansion

1 Introduction

Quantity and quality of training data are crucial regarding the usefulness of a classifier. In scenarios where data collection is limited or expensive, e.g., compute-intensive simulations for design-space exploration [11, 12, 25], the data set collected often is small and biased, i.e., does not represent the real data distribution well. In this case, data synthesis is promising to extend the initial data sample. However, synthesis is a hard problem when the domain of the data is unknown, potentially unbound, and of high dimensionality, since randomly generating observations in the data space is inefficient. At the same time, it is also challenging for a user to extend a high dimensional data set by hand. To keep the user effort low, active learning has been proposed. Here, a so-called query synthesis strategy generates artificial observations for which the user or the simulation then provides the labels, aka. feedback. In this article, we

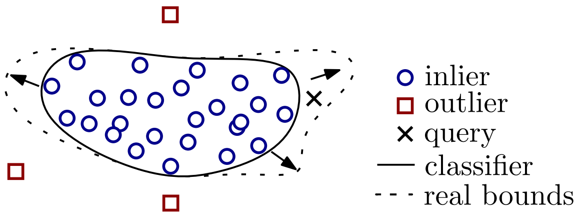


Figure 1: Domain expansion with query synthesis.

focus on expanding the classifier knowledge beyond an initial sample of inliers that form a single connected region in an unbound data space. We call this the *domain expansion problem.* Figure 1 is an illustration. We start with a small initial sample of mostly inliers and seek to expand the classifier boundaries of a one-class classifier towards the real boundaries.

Existing approaches for query synthesis are for the multi-class setting and require observations from all classes [21, 38]. They cannot synthesize queries in directions where no observations exist, as is the case with domain expansion. Next, methods for design space exploration [11, 12, 25] and reliability-based design optimization [26] use the prediction uncertainty of a binary classifier for query generation. But this uncertainty may not be obtainable from a one-class classifier. So we target at a method for domain expansion that performs query synthesis in one-class active learning.

1.1 Challenges Query synthesis in the one-class settings is difficult for three reasons.

High dimensionality Existing one-class query strategies use the unlabeled observations as query candidates and query the one with the highest expected information gain [35]. Without such observations, the volume of a high-dimensional data space is too large for a random candidate generation. This calls for a more selective candidate placement, e.g., near the current observations.

Data Set Pollution During query synthesis, artificially generated queries and their feedback are inserted into the original data set. This affects the data density and the neighborhoods. To illustrate, querying in the same region several times increases the data density where it may be low in reality. Yet several one-class

[*]Karlsruhe Institute of Technology (KIT), Germany, `adrian.englhardt@kit.edu` , `klemens.boehm@kit.edu`

classifiers such as Nearest Neighbor Description [31] and pool-based query strategies [16, 17, 19, 40] rely on these data characteristics. Query synthesis may then misguide the classifiers and query strategies.

Evaluation standards Evaluation standards from pool-based one-class active learning setting do not carry over to query synthesis. On the one hand, an evaluation needs a simulated annotator, a so-called oracle. With query synthesis, it is unclear how to simulate an oracle that can provide an answer for an arbitrary query. On the other hand, one must compare a new query synthesis strategy to a baseline. But the existing baseline to select a random query [19] is infeasible, and sampling from a potentially unbounded data space is ineffective. In this article we study how to construct a more realistic baseline, by deriving boundaries from the available data.

1.2 Contributions This article features an approach for query synthesis for one-class active learning. It performs domain expansion based on initial observations from one class. Our contributions are as follows:

Framework (C1) We generalize pool-based one-class active learning and propose a framework for query synthesis in one-class active learning (SYNOCAL). The framework is defined by the initial labels, a one-class classifier and a query synthesis strategy. We frame query synthesis as an optimization problem. This allows for efficient query synthesis, given any query strategy that quantifies the expected information gain of a candidate. We then use meta-heuristics to solve this problem even for high-dimensional data. The framework addresses the challenges *one-class scenario* and *high dimensionality*.

Domain Expansion Strategy *DES* (C2) We propose a new query-synthesis strategy to expand the classifier knowledge beyond the initial, potentially biased, data sample by leveraging work on adaptive data shifting [39]. Our novel strategy allows to learn in areas of uncertainty where no observations exist.

Evaluation standards (C3) We propose evaluation standards for one-class query synthesis. We introduce three ways how to define an oracle using synthetic and real world data. Additionally, we derive two realistic baselines strategies for query synthesis from existing artificial outlier generation algorithms [1, 13].

In the evaluation, we perform extensive experiments for different domain expansion problems. We show that our new query-synthesis strategy outperforms all alternatives in settings with few inlier labels. We make the implementation of the framework to reproduce our results publicly available.[1]

2 Related Work

We review work on multi-class query synthesis and one-class active learning. We also discuss artificial outlier generation, adversarial attacks, and experimental design related to one-class query synthesis.

Multi-class Query Synthesis Research has proposed several approaches for query synthesis in a binary classification setting. One uses observations pairs that belong to opposite classes to synthesize queries along the decision boundary [38]. Two others choose queries that shrink the version space of potential decision boundaries [2, 10]. Another option is to first cluster the data, then synthesize a query between the centroids of class clusters [20, 21]. All these approaches require negative examples, which may be missing in a one-class setting. Next, previous work features handcrafted query synthesis strategies, e.g., for biological experiments [23, 24], that do not generalize to other domains.

One-Class Active Learning All existing one-class approaches are pool-based query strategies. They compute the expected information gain for all unlabeled observations and then query the best one. Following the categorization in [35], there are three types of one-class query strategies. The first type, data-based query strategies, selects queries independent of the classifier only based on data characteristics such as densities [16, 17] The other two types of strategies are based on a one-class classifier that learns a decision boundary, e.g., support vector data description (SVDD) [33]. Model-based strategies query the unlabeled observation closest to [18] or farthest from [4] the learned decision boundary. Without unlabeled observations, it is unclear what a query far away from the decision boundary would be. We will use the idea of querying observations that lie on the decision boundary and combine it with query synthesis. The last category of query strategies are hybrid strategies that combine the distance to the decision boundary with neighborhood information [19, 40]. Data-based and hybrid query strategies are not suitable for our setting because *data set pollution* affects the densities and neighborhoods.

Artificial Outliers Another area where observations are synthesized in the context of one-class learning is artificial outlier generation. Artificial outliers are used to transform an unsupervised problem to a supervised one [1, 3, 13]. They allow to balance a classification problem for, say, outlier detection. One can then use traditional binary classifiers such as a SVM [3]. Literature also suggests to tune the hyper-parameters of

[1] `https://www.ipd.kit.edu/des`

one-class classifiers with synthetic outliers [32, 37, 39]. In both use cases the goal is to train a classifier with a high outlier detection rate. However, algorithms for artificial outlier generation are not designed for active learning since they are independent of the classifier.

Adverserial Attacks Query synthesis is also used in the context of adversarial attacks. Here, an attacker seeks to evade the detection by a classifier while changing his malfeasance only minimally. Existing approaches try to reconstruct the decision boundary [27, 30] or search for an attack instance close to the desired malfeasance that the classifier does not detect [28]. Adversarial query strategies assume a fixed classifier, i.e., they do not consider user feedback.

Design of Experiments Engineers perform query synthesis to explore the experimental design space. The literature distinguishes between *exploration* of this space, i.e., finding new feasible regions, and *exploitation*, the refinement in areas with existing observations [11]. Several approaches take a bounded design space and perform adaptive sampling [5, 8, 25, 26, 29]. One other approach exists that explores an unbounded design space [11]. Existing work then trains a surrogate model on the obtained labels – commonly a Gaussian process classifier [8, 11, 25, 26] or a SVM [5, 29]. Both models are not applicable since they are binary classifiers and require labels of both classes.

3 Framework

In this section we present our *framework* (C1). We formalize one-class query synthesis and present several query strategies that serve as baselines later. The framework is the fundament for our novel domain expansion strategy. Yet it is general and facilitates research on one-class query synthesis strategies.

Definition 3.1. (Query synthesis strategy) *Given a data set X with labels $\mathcal{L}_{in}$, $\mathcal{L}_{out}$ and a classifier $\mathcal{C}$, a query synthesis strategy QSS is a function of type $QSS : \mathcal{C}, \mathcal{U}, \mathcal{L}_{in}, \mathcal{L}_{out} \rightarrow \mathcal{Q}$ where $\mathcal{Q} \in \mathcal{X}$ is an artificial query for feedback collection.*

Definition 3.2. (Query synthesis) *Query synthesis is a method to improve the classification by a classifier $\mathcal{C}$ on data X by acquiring feedback from an oracle $\mathcal{O}$ on queries generated by a query synthesis strategy QSS.*

In a one-class setting, classifier and query synthesis strategy must cope with imbalanced class distributions. To this end, we propose a new framework called SYNOCAL to perform query SYNthesis in One-Class Active Learning. SYNOCAL consists of three elements: (1) active learning scenario, (2) one-class classifier and (3) query synthesis strategy.

Notation $\mathcal{X} \subseteq \mathbb{R}^d$ is the data space with d attributes. $X \in \mathbb{R}^{n \times d}$ is a data set with n observations $x_i \in X$. The ground truth labels of an observation are "inlier" or "outlier". During active learning an observation can either be in the unlabeled set $\mathcal{U}$ or in the labeled set of inliers $\mathcal{L}_{in}$ or outliers $\mathcal{L}_{out}$ where $\mathcal{U}, \mathcal{L}_{in}, \mathcal{L}_{out} \subseteq X$.

3.1 Active Learning Scenario

Previous research on one-class active learning relies on assumptions that affect the interaction of a user with the active learning system or confine the choice of the classifier and the query strategy [35, 36]. The active learning scenario specifies which label information is available initially. There are three categories:

- Unsupervised: All observations are unlabeled: $\mathcal{U} = X$ with $\mathcal{L}_{in} = \mathcal{L}_{out} = \emptyset$
- Semi-supervised: Inliers and unlabeled observations are present, optionally some outliers: $\mathcal{L}_{in} \cup \mathcal{L}_{out} \cup \mathcal{U} = X$ with $\mathcal{L}_{in} \neq \emptyset \wedge \mathcal{U} \neq \emptyset \wedge |\mathcal{L}_{out}| \ll |\mathcal{L}_{in}|$
- Supervised: Only labeled observations are available with most of them being inliers: $\mathcal{L}_{in} \cup \mathcal{L}_{out} = X$ with $\mathcal{L}_{in} \neq \emptyset \wedge \mathcal{U} = \emptyset \wedge |\mathcal{L}_{out}| \ll |\mathcal{L}_{in}|$

Pool-based query strategies only work with the unsupervised and semi-supervised scenario, while query synthesis works with any setup.

3.2 One-Class Classifier

The second element of our framework is the one-class classifier (OCC). A one-class classifier outputs a decision function, as follows:

Definition 3.3. (Decision Function) *A decision function f is a function of type $f : \mathcal{X} \rightarrow \mathbb{R}$. An observation belongs to the outlier class if $f(x) > 0$ and to the inlier class otherwise.*

3.2.1 Requirements

We derive two requirements that a one-class classifier must meet for query synthesis:

- *Semi-supervised:* The classifier must be able to learn from feedback. So the classifier must be semi-supervised, to make use of labeled and potentially unlabeled observations.
- *Boundary-based:* To avoid the *data set pollution problem*, the classifier must not use densities or neighborhoods. This calls for a classifier that learns tight enclosing boundaries around the inliers.

3.2.2 One-Class Classifiers Choice

Research proposed a variety of one-class classifiers [22] with many specific variants, like classification with weights [41] or in subspaces [34]. Following our requirements this leaves us with classifiers based on Support Vector Data Description (SVDD) [33]. A SVDD-based classifier solves a Minimum Enclosing Ball (MEB) optimization problem and outputs a hyper-sphere with a center a and radius R. To give the classifier more flexibility, the data is usually transformed into a reproducing kernel Hilbert space with a function ϕ. The decision function for a SVDD-based classifier is:

$$f(x) = \|\phi(x) - a\| - R. \tag{3.1}$$

The original SVDD is unsupervised and therefore cannot learn from feedback. SVDDneg is an extension which enforces that labeled outliers fall outside of the hyper-sphere [33]. In addition to this, SSAD modifies the objective function and adds additional constraints so that all labeled observations fall on the correct side of the hyper-sphere [19]. SVDDneg and SSAD meet all our requirements for one-class query synthesis.

3.3 Query Synthesis Strategy The third element of our framework is the query synthesis strategy. Such a strategy uses the classifier, available observations and labels to generate an artificial query. We differentiate between direct and indirect *QSS*.

Direct QSS The first type directly synthesizes the query given the classifier, available observations and labels. Baseline strategies that randomly generate a query are direct strategies. To define these baselines, we follow the approaches for artificial outlier generation [1, 13, 32] and bound the data space with a hyper-rectangle that encloses all observations. A hyper-rectangle H is defined by two boundary vectors $a, b \in \mathbb{R}^d$, usually the minima and maxima along each dimension. H_ε stands for such an expanded hyper-rectangle where $\varepsilon \in \mathbb{R}^+$ is the expansion, e.g., $H_{0.1}$ for a 10% expansion. This gives way to our two baselines.

- $DQSS_{\text{rand}}$: Samples a random query $\mathcal{Q} \in H_\varepsilon$.
- $DQSS_{\text{rand-o}}$: Samples a query $\mathcal{Q} \in H_\varepsilon$ with $f(\mathcal{Q}) > 0$, i.e., classified as outlier by $\mathcal{C}$.

Indirect QSS The second type on the other hand first defines the expected information gain for an arbitrary observation which is then optimized:

Definition 3.4. (Expected information gain) *Given a classifier $\mathcal{C}$ and label sets $\mathcal{U}$, $\mathcal{L}_{in}$, $\mathcal{L}_{out}$, the expected information gain is a function $x \mapsto \tau(x, \mathcal{C}, \mathcal{U}, \mathcal{L}_{in}, \mathcal{L}_{out})$ that maps an arbitrary $x \in \mathcal{X}$ to $\mathbb{R}$.*

τ quantifies the expected information gain, i.e., how valuable an arbitrary observation $x \in \mathcal{X}$ is. To illustrate, the expected information gain can be high in areas close to the decision boundary of a classifier. As $\mathcal{C}$, $\mathcal{U}$, $\mathcal{L}_{in}$ and $\mathcal{L}_{out}$ are fixed in a given active learning iteration, we only write $\tau(x)$. One can then optimize $\tau(x)$ to obtain the optimal synthetic query:

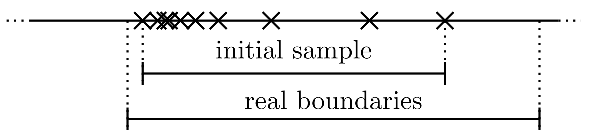


Figure 2: 1-dimensional domain expansion problem.

Definition 3.5. (Query synthesis optimizer) *Given an* indirect *query synthesis strategy with a function $\tau(x)$, a query synthesis optimizer (QSO) yields the optimal query $\mathcal{Q}$ by computing $\mathcal{Q} = \arg\max_{x \in \mathcal{X}} \tau(x)$.*

There is a variety of optimization algorithms, and the optimal choice depends on $\tau(x)$. Preliminary experiments of ours have shown that the derivative-free metaheuristic evolutionary optimizer DXNES [15] offers high solution quality in reasonable compute time.

$IQSS_{DB}$: Querying observations close to the decision boundary [18] is a pool-based strategy that we can adapt as indirect query synthesis strategy. Given the decision function f of a classifier, $IQSS_{\text{DB}}$ defines $\tau_{\text{DB}} = -|f(x)|$. The strategy gives a high expected information gain to queries on the margin between the two classes where the classification uncertainty is highest.

4 Domain Expansion

This section deals with the *domain expansion* problem. We first formalize it and then present our novel query synthesis strategy (C2).

4.1 Formalization We define $\mathcal{C}^*$ as the optimal one-class classifier for the data space $\mathcal{X}$. One starts with a sample of inliers X drawn from $\mathcal{X}$ and seeks to learn a classifier $\mathcal{C}$ that matches $\mathcal{C}^*$. To do so, we use query synthesis and perform one-class active learning. The challenge now is the definition of a query synthesis strategy to achieve this. We split this problem into two subproblems.

Identification of exploration direction (IED) The first challenge is the identification of an exploration direction. Figure 2 gives a 1-dimensional example with an initial sample and the real boundaries. Here, one might prefer exploring to the left, due to the higher density of the sample in this area. However, first exploring to the right is more beneficial. The space uncovered by the initial sample is larger to the right than to the left when looking at the hidden real boundaries. Generally, the initial sample does not give any information on the direction to choose. This is because it can be biased and may not represent the actual distribution of the data. Initially, one therefore has to rely on guessing. In high dimensional data spaces where the boundaries of the initial sample match the real boundaries well, and only a few directions would yield an improved boundary, it becomes more difficult to guess the "improving" directions. So more queries are necessary to check all directions. When a user has labeled an outlier in a direction d_{out}, one should either explore in different areas or decrease the vector length of d_{out} to query between the outlier and the known inliers.

Identification of exploration magnitude (IEM) One has to define how far to explore in a given direction. Initially one only has inlier samples. One option again is to guess the exploration magnitude. But different value ranges require different magnitudes. To illustrate, exploring a 1-dimensional data set with a hidden value range of $[1, 100]$ and initial sample value range of $[5, 80]$ with an exploration magnitude of 0.001 requires many queries. So one should infer the magnitude from the initial sample. In the case where one continuously retrieves inlier feedback for queries in one direction, one could gradually increase the magnitude to explore the direction more quickly. When outlier labels become available, one should decrease the magnitude in the outlier direction to query between the outlier and the known inliers.

4.2 Our Solution We now present our solution and say how we address the two subproblems. We propose a new domain expansion strategy (DES) that performs query-synthesis. We briefly describe the intuition behind our approach first and then present its components.

Intuition Our new query strategy is an indirect query synthesis strategy with an expected information gain function. Following our previous discussion, all exploration directions initially have the same weight. Since the dimensionality and the shape of the inlier region vary between data sets, we derive these directions directly from the SVDD-based classifier. Without loss of generality, all exploration directions are orthogonal to the decision boundary. When restricting them to length ε, they form a hull around the initial sample. This hull corresponds to the decision boundary that is shifted away from the initial sample by some ε. Additionally, we must consider any labeled outliers and decrease the exploration magnitude in directions with outliers, i.e., shift the boundary less far. To this end, we propose a new one-class classifier SVDDnegEps that learns such a shifted decision boundary while respecting outlier labels. Then we propose a parametrization method to compute the exploration magnitude, i.e., the shift ε. Finally, we introduce a method to quickly prune the search space of all possible exploration directions and magnitudes by decreasing the expected information gain in areas with outliers. This pruning enforces query diversity over multiple active learning cycles.

SVDDnegEps Our novel one-class classifier *SVDDnegEps* learns a shifted decision boundary where outliers stay outside of the hyper-sphere. SVDDnegEps enforces an extended boundary by injecting the exploration magnitude, i.e., the shift amount ε, into the constraints of the SVDDneg optimization problem:

$$\begin{aligned} \min_{R,a,\xi} \quad & R^2 + C_1 \sum_i \xi_i + C_2 \sum_l \xi_l \\ \text{s.t.} \quad & \|\Phi(x_i) - a\|^2 + \varepsilon \le R^2 + \xi_i,\ \forall i \\ & \|\Phi(x_l) - a\|^2 \le R^2 - \xi_l,\ \forall l \\ & \xi_i, \ge 0,\ \forall i \\ & \xi_l, \ge 0,\ \forall l \end{aligned} \tag{4.2}$$

The index i refers to observations in $\mathcal{U} \cup \mathcal{L}_{in}$ and l to outliers, ξ_i and ξ_l are the corresponding slack variables and $C_1, C_2 \in [0, 1]$ the cost parameters. After solving the problem, we have a fixed center a and radius R that define the enclosing hyper-sphere. We can then use the decision function Equation 3.1.

Parametrization We use the artificial outlier generation approach from [39] to infer ε. The approach generates artificial outliers to tune the hyper-parameters of a SVDD. The generation process places artificial outliers around the data by shifting boundary observations along their negative data density estimated from the neighborhoods. So the approach adapts to arbitrary value ranges. Our idea is to use these artificial outliers to quantify the maximum magnitude how far one should explore. We can measure the distance of these outliers to the decision boundary of a OCC with the decision function f of the classifier. Given f and artificial outliers X_{out}, the shift is:

$$\varepsilon = \max_{x_i \in X_{\text{out}}} f(x_i) \tag{4.3}$$

This bounds the maximum exploration to the farthest artificial outlier generated from the initial sample.

Domain Expansion Strategy Our novel query strategy $IQSS_{\text{DES}}$ is similar to the decision boundary strategy $IQSS_{\text{DB}}$, except that it queries on the shifted decision boundary of a SVDDnegEps. f_ε stands for this decision function, to make the difference to f explicit. The expected information gain of the $IQSS_{\text{DES}}$ is:

$$\tau_{\text{DES}} = -|f_\varepsilon(x)|. \tag{4.4}$$

We can then use our framework to perform query synthesis with an optimizer.

Search space pruning In high dimensional data spaces, one should punish areas with negative feedback and encourage exploration elsewhere. To this end, we propose a method to quickly prune the search space of the exploration directions and magnitudes. This extension is general and works with an arbitrary indirect query synthesis strategy (IQSS) that defines an expected information gain. The idea is to automatically reduce the expected information gain in areas with outliers. To this end, we train a binary SVM on the available data. By using a binary SVM, one can quantify the distance to the decision boundary in the same kernel space as for the SVDD-based classifier. So no value-range adjustment is needed, and we can directly combine the distance to the SVM and to the OCC. The decision function for a SVM is f_{SVM}. Again, an outlier x yields $f_{\text{SVM}}(x) > 0$, and an inlier x gives a value $f_{\text{SVM}}(x) \leq 0$. Given a IQSS with expected information gain τ, we then define a combined expected information gain function τ^*:

$$\tau^*(x) = \tau(x) - \max(f_{\text{SVM}}(x), 0) \tag{4.5}$$

This modification of τ punishes areas with negative feedback (IED) and reduces the exploration magnitude in the direction of outliers (IEM).

4.3 Example

We now illustrate how our query synthesis strategy with search space pruning, dubbed $IQSS^*_{\text{DES}}$, works, cf. Figure 3. Figure 3a shows the start of the active learning cycle. Here, the black line is the decision boundary fitted by a SVDDneg, and the dashed line is the oracle, i.e., the target boundary that we want to learn. The initial sample does not cover the full valid space and misses a large area to the right. The heatmap indicates the expected information gain – the darker, the higher the gain. After five queries, we have expanded the boundary and have acquired the first outlier. The pruning then punishes the area around the outlier, see Figure 3b, and the query strategy starts exploring other areas. The state after 15 queries is visualized in Figure 3c. Our method has already approximated the real decision boundary quite well.

5 Evaluation

Previous work uses benchmark data sets with a ground truth and performs pool-based active learning. However, an evaluation is more difficult in the context of query synthesis because the oracle must provide a label for arbitrary queries. We see three options how to evaluate one-class query synthesis.

Synthetic Data The first option relies on synthetic data. Here, one defines a function that acts as the oracle. For instance, one can use a multivariate distribution. The oracle function then labels an observation as outlier if the density falls under a threshold. Additionally, the distribution allows to generate an arbitrary volume of test data to evaluate the classifier.

Ground Truth Fitting The second option relies on existing one-class data sets. One can fit a classifier to the ground truth data to obtain an oracle. The fitted classifier can then serve as an oracle to answer arbitrary queries in the active learning cycle. Example classifiers are SVDD or also basic classifiers such as SVM or kNN classifier. The are two problems with this approach. First, the oracle is limited by the capabilities of what the underlying classifier can learn. So one may not achieve a perfect fit to the data, and queries may yield wrong feedback. Second, one-class outlier benchmark sets are unbalanced. The oracle classifier has high uncertainty in these sparse areas. Additionally, we also have a very limited number of outliers for testing. Active learning may improve a classifier, but there is no test data to reflect this improvement.

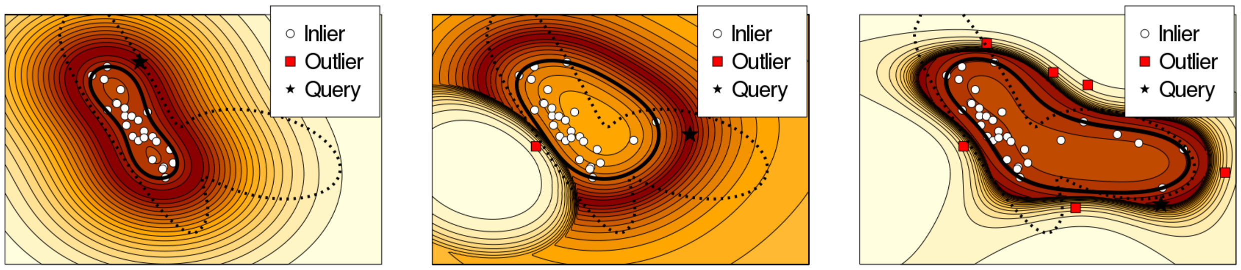
Hybrid Query Synthesis The third option avoids crafting an oracle by only allowing queries for existing observations. With query synthesis, one can use a *hybrid approach*: One first uses any strategy to synthesize a query $\mathcal{Q}$. Then the observation closest to $\mathcal{Q}$ under some distance function *dist* becomes the query:

$$\mathcal{Q}^* = \underset{x \in X}{\arg\min}\ dist(x, \mathcal{Q}) \tag{5.6}$$

This hybrid approach generally avoids the problem of awkward queries [6] and has already been used to evaluate multi-class query synthesis [38]. However, we see a problem in the one-class setting with this option. The selected unlabeled observation may be far away from the synthesized query due to data sparsity. In this case, a query strategy may have preferred a completely different observation when computing the expected information gain on the unlabeled observations, compared to the one chosen by the hybrid approach.

6 Experiments

In this section we evaluate our framework and our new query synthesis strategy on different domain expansion problems. In the first part, we use synthetic data of different dimensionality. In the second part, we assess the performance of our method on common real-world data sets for outlier detection [9]. We have implemented the query synthesis framework and the benchmark setup in Julia [7]. Our implementation, the raw results of all settings and notebooks to reproduce experiments and evaluation are publicly available at `https://www.ipd.kit.edu/des`.



(a) Iteration 1 (b) Iteration 5 (c) Iteration 15

Figure 3: Expanding the decision boundary of a SVDDneg with our new $IQSS^*_{\text{DES}}$ with search space pruning.

Data and Oracle We use synthetic and real-world data sets for our experiments. We present the data sets and how we define the oracle.

Synthetic data: Gaussian mixtures allow to flexibly generate synthetic domain expansion problems. We generate 100 random Gaussian Mixtures for each combination of 3, 5 and 7 components and 2, 4, 6, 8 and 10 dimensions. To bias the initial sample, we then choose the initial sample only from one of the components. Given the Gaussian mixture with density function $p(x)$, we use a threshold $t = 0.1$ to define the oracle: Observations are inliers if $p(x) \geq t$ and outliers otherwise. To evaluate the classifier, we generate 1000 inliers and outliers each in addition to the initial sample.

Real-world data: For the real-world experiments we use the publicly available outlier benchmark data sets from [9]. We use the normalized and deduplicated version of 15 different data sets. They have different sizes (80–7129 observations) and outlier ratios (4–75%). In order to limit the search space, we perform feature selection to extract the 5 most meaningful features using mutual information. This is in line with the discussion in Section 4 where we argue that guessing an exploration direction at the start of the learning becomes more difficult with increasing dimensionality. To choose the initial sample, we perform a neighborhood walk: We first randomly select one observation and iteratively add the next nearest neighbor to the initial sample until we reach the desired sample size. In this way, we bias the initial sample into one area. We run 10 resamples of the initial sample for each data set. For the oracle we perform *ground truth fitting*. Preliminary experiments have shown that a binary SVM performs better as an oracle than a one-class classifier when using all labels. We additionally generate synthetic inliers and outliers with the method proposed in [39] to tune the kernel parameter of the oracle SVM with cross-validation.

Parametrization To simulate a small biased sample to expand from, we set the number of initial observations to 25. We then run 100 active learning iterations with our framework. As a model we use the SVDDneg [33] with a hard margin $C_1, C_2 = 1.0$ and tune the kernel parameter with the method proposed in [39]. We use the same parameters for the SVDD-negEps of $IQSS_{\text{DES}}$. For the baseline query strategies $DQSS_{\text{rand}}$ and $DQSS_{\text{rand-o}}$, we set the hyper-rectangle expansion to $\varepsilon = 0.1$ as in [1]. For the indirect query synthesis strategies we use *distance-weighted exponential natural evolution strategy* [15] (DXNES) with the default parameters implemented in [14]. We fix the optimization boundaries to an extended hyper-rectangle $H_{1.0}$ with 100% expansion beyond the labeled inliers.

Evaluation metrics To evaluate the results, we calculate the Matthews Correlation Coefficient (MCC) to compare the classifier prediction for the test data with the ground truth for each iteration. MCC is a metric especially suitable for imbalanced data sets. We then calculate the *End Quality* (EQ) [35] to quantify the performance of the classifier after the active learning.

Results We now present and discuss the results of our experiments to show the superiority of our novel query strategy on different domain expansion problems.

Synthetic Data We first run our framework with two baselines $DQSS_{\text{rand}}$, $DQSS_{\text{rand-o}}$ and indirect query synthesis strategies $IQSS_{\text{DB}}$, $IQSS_{\text{DES}}$ and with search space pruning $IQSS^*_{\text{DES}}$ on the synthetic data. Figure 4 shows the end quality for different dimensionalities. Generally, our framework allows to improve the classifier from a small initial sample even in the absence of unlabeled observations. The end quality decreases with increasing data dimensionality. This is because finding a helpful exploration direction becomes more difficult and thus requires more queries – here we are using a fixed number of 100 iterations. Our proposed approach with search space pruning $IQSS^*_{\text{DES}}$ outperforms or is at least on par with all other strategies. The baseline strategy $DQSS_{\text{rand-o}}$ proposed earlier yields competitive results. This is similar to the results in [35] where a random baseline for pool-based

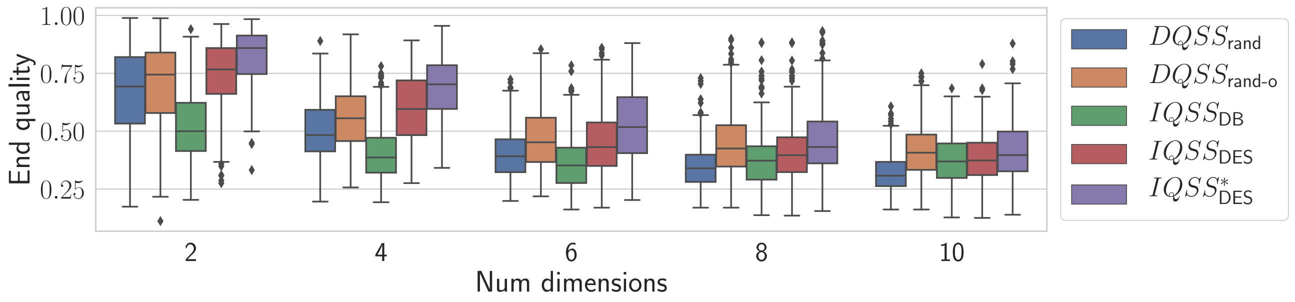


Figure 4: End quality for synthetic data sets with different dimensionality.

Data set	rand	rand-o	DB	DES	DES*
Annthyroid	0.05	0.07	0.03	0.04	**0.08**
Arrhythmia	0.18	0.24	0.17	0.18	**0.27**
Cardio	0.09	0.14	0.06	0.07	**0.20**
Glass	0.13	0.16	0.11	0.20	**0.24**
HeartDisease	0.22	0.27	0.18	0.26	**0.28**
Hepatitis	0.28	**0.31**	0.27	0.19	0.21
InternetAds	0.22	0.65	0.13	0.14	**0.73**
Ionosphere	0.22	0.26	0.22	0.27	**0.49**
PageBlocks	0.10	0.17	0.05	0.06	**0.22**
Parkinson	0.62	0.62	**0.64**	0.60	0.42
Pima	0.14	0.13	**0.15**	0.14	0.14
SpamBase	0.19	**0.46**	0.13	0.12	0.44
Stamps	0.15	**0.26**	0.11	0.14	0.22
WPBC	-0.03	-0.06	-0.02	-0.05	**0.01**
Wilt	-0.08	-0.06	-0.06	**-0.05**	-0.08

Table 1: Median end quality on real world data sets.

one-class learning outperforms other strategies on one third of the data sets. Querying directly on the decision boundary with $IQSS_{\text{DB}}$ does not yield any improvement. $IQSS_{\text{DB}}$ mostly queries inliers near the initial sample and does not gain insights beyond the sample.

Real-World Data We run the same setup on the real-world data sets. Table 1 shows the median end quality for all the proposed query synthesis strategies on different data sets. The end quality with active learning varies from data set to data set. For the WPBC and Wilt data set no method yields any considerable improvement which results in a low end quality. Except for them, our proposed method $IQSS^*_{\text{DES}}$ achieves the best results overall, winning in 9 out of 13 data sets. Model quality significantly increases from the initial set with $IQSS^*_{\text{DES}}$. Similarly to the synthetic evaluation, our method benefits from search space pruning. One can see this by comparing the results of $IQSS_{\text{DES}}$ and $IQSS^*_{\text{DES}}$.

7 Conclusions

This article studies the domain expansion problem where one seeks to improve a one-class classifier by extending a small initial data sample with additional training data. To this end, we have proposed a general framework that frames query synthesis in the one-class setting as an optimization problem. We then have proposed a novel domain expansion strategy that explores the data space and does away with the bias of small samples. The method includes a pruning of the considered exploration space that diversifies the queries considerably. Evaluating query synthesis strategies requires an oracle that can provide an answer for arbitrary queries. In this article, we derive three ways to use synthetic and real-world data to simulate such an oracle. Comprehensive experiments on synthetic and real-world domain expansion problems demonstrate that our method indeed expands the knowledge of a classifier beyond a biased sample and outperforms realistic baselines.

Acknowledgments

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

References

[1] Naoki Abe, Bianca Zadrozny, and John Langford. “Outlier detection by active learning”. In: *SIGKDD*. ACM. 2006.

[2] Ibrahim M Alabdulmohsin, Xin Gao, and Xiangliang Zhang. “Efficient Active Learning of Halfspaces via Query Synthesis.” In: *AAAI*. 2015.

[3] András Bánhalmi, András Kocsor, and Róbert Busa-Fekete. “Counter-Example Generation-Based One-Class Classification”. In: *ECML*. Springer. 2007.

[4] V. Barnabé-Lortie, C. Bellinger, and N. Japkowicz. “Active Learning for One-Class Classification”. In: *ICMLA*. IEEE. 2015.

[5] Anirban Basudhar and Samy Missoum. “Adaptive explicit decision functions for probabilistic design and optimization using support vector machines”. In: *Computers & Structures* 86.19–20 (2008).

[6] Eric Baum and Kenneth Lang. “Query learning can work poorly when a human oracle is used”. In: *IJCNN*. 1992.

[7] Jeff Bezanson et al. “Julia: A fresh approach to numerical computing”. In: *SIAM Review* (2017).

[8] Brent Bryan et al. “Active learning for identifying function threshold boundaries”. In: *NIPS*. 2006.

[9] Guilherme Campos et al. “On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study”. In: *Data Min Knowl Disc* (2016).

[10] Lin Chen, Seyed Hamed Hassani, and Amin Karbasi. “Near-Optimal Active Learning of Halfspaces via Query Synthesis in the Noisy Setting.” In: *AAAI*. 2017.

[11] Wei Chen and Mark Fuge. “Active expansion sampling for learning feasible domains in an unbounded input space”. In: *Struct Multidisc Optim* (2017).

[12] Wei Chen and Mark Fuge. “Beyond the known: Detecting novel feasible domains over an unbounded design space”. In: *J Mech Des* 139.11 (2017).

[13] Chesner Désir et al. “One class random forests”. In: *Pattern Recognition* 46.12 (2013).

[14] Robert Feldt. *BlackBoxOptim.jl*. `https://github.com/robertfeldt/BlackBoxOptim.jl`. 2018.

[15] Nobusumi Fukushima et al. “Proposal of distance-weighted exponential natural evolution strategies”. In: *Congress of Evolutionary Computation (CEC)*. 2011.

[16] A. Ghasemi et al. “Active Learning from Positive and Unlabeled Data”. In: *ICDM Workshop*. 2011.

[17] A. Ghasemi et al. “Active one-class learning by kernel density estimation”. In: *MLSP Workshop*. 2011.

[18] Nico Görnitz et al. “Active Learning for Network Intrusion Detection”. In: *AiSec Workshop*. ACM, 2009.

[19] Nico Görnitz et al. “Toward Supervised Anomaly Detection”. In: *JAIR* (2013).

[20] Xuelei Hu, Liantao Wang, and Bo Yuan. “Querying representative points from a pool based on synthesized queries”. In: *IJCNN*. 2012.

[21] Ajay Jayant Joshi. “Image classification with minimal supervision”. In: (2011).

[22] Shehroz S. Khan and Michael G. Madden. “One-class classification: taxonomy of study and review of techniques”. In: *Know Eng Rev* (2014).

[23] Ross D King et al. “The automation of science”. In: *Science* 324.5923 (2009).

[24] Ross D King et al. “Functional genomic hypothesis generation and experimentation by a robot scientist”. In: *Nature* 427.6971 (2004).

[25] Brad J Larson and Christopher A Mattson. “Design space exploration for quantifying a system model’s feasible domain”. In: *J Mech Des* 134.4 (2012).

[26] Tae Hee Lee and Jae Jun Jung. “A sampling technique enhancing accuracy and efficiency of metamodel-based RBDO: Constraint boundary sampling”. In: *Computers & Structures* 86.13-14 (2008).

[27] Daniel Lowd and Christopher Meek. “Adversarial learning”. In: *SIGKDD*. 2005.

[28] Blaine Nelson et al. “Query strategies for evading convex-inducing classifiers”. In: *JMLR* 13 (2012).

[29] Yi Ren and Panos Y Papalambros. “A design preference elicitation query as an optimization process”. In: *J Mech Des* 133.11 (2011).

[30] Tegjyot Singh Sethi and Mehmed Kantardzic. “Data driven exploratory attacks on black box classifiers in adversarial domains”. In: *Neurocomputing* 289 (2018).

[31] David M. J. Tax. “One-class classification: Concept learning in the absence of counter-examples”. In: (2002).

[32] David M. J. Tax and Robert P. W. Duin. “Uniform Object Generation for Optimizing One-class Classifiers”. In: *JMLR* (2002).

[33] David M. J. Tax and Robert P.W. Duin. “Support Vector Data Description”. In: *Machine Learning* (2004).

[34] Holger Trittenbach and Klemens Böhm. “One-Class Active Learning for Outlier Detection with Multiple Subspaces”. In: *CIKM*. 2019, pp. 811–820.

[35] Holger Trittenbach, Adrian Englhardt, and Klemens Böhm. “An Overview and a Benchmark of Active Learning for One-Class Classification”. In: *arXiv preprint arXiv:1808.04759* (2018).

[36] Holger Trittenbach, Adrian Englhardt, and Klemens Böhm. “Validating One-Class Active Learning with User Studies–a Prototype and Open Challenges”. In: *ECML PKDD Workshop*. 2019, p. 17.

[37] Chi-Kai Wang et al. “A novel approach to generate artificial outliers for support vector data description”. In: *Industrial Electronics*. IEEE. 2009.

[38] Liantao Wang et al. “Active learning via query synthesis and nearest neighbour search”. In: *Neurocomputing* 147 (2015).

[39] Siqi Wang et al. “Hyperparameter selection of one-class support vector machine by self-adaptive data shifting”. In: *Pattern Recognition* (2018).

[40] Lili Yin, Huangang Wang, and Wenhui Fan. “Active learning based support vector data description method for robust novelty detection”. In: *Knowl-Based Syst* (2018).

[41] Yong Zhang et al. “Fault classifier of rotating machinery based on weighted support vector data description”. In: *Expert Syst Appl* 36.4 (2009), pp. 7928–7932.