Finding the Sweet Spot: Batch Selection for One-Class Active Learning

Adrian Englhardt* Holger Trittenbach* Dennis Vetter* Klemens Böhm*

Abstract

Active learning methods collect annotations in the form of class labels, often from human experts, to improve upon some classification task. In many cases, one can collect annotations for a batch of observations at a time, e.g., when several annotators are available. This can make the annotation process more efficient, both regarding human and computational resources. However, selecting a good batch is difficult. It requires to understand several trade-offs between the costs of classifier training, batch selection, annotation effort, and classification accuracy. For one-class classification, a very important application of active learning, batch selection has not received any attention in the literature so far. In this article, we strive to find a sweet spot between the costs of batch-mode learning and classification accuracy. To this end, we first frame batch selection as an optimization problem. We then propose several strategies to identify good batches, discuss their properties, and evaluate them on real-world data. A core result is that a sweet spot indeed exists, with active learning costs reduced by up to an order of magnitude compared to a sequential procedure, without sacrificing accuracy.

Keywords one-class classification, active learning, batch queries

1 Introduction

Active learning is the machine-learning paradigm to ask an oracle for auxiliary information, to increase classification quality. The oracle often is a human annotator who provides class labels for observations. Annotations may also be of different nature, like the outcome of a simulation or experiment [4]. An important application of active learning is anomaly detection with one-class classifiers [1, 13–15, 33, 40]. Here, the oracle provides a binary class label – "inlier" or "outlier". Feedback typically is *sequential*, i.e., once the oracle has provided a class label for a single observation, the classifier is retrained. But there are cases when the sequential mode is unfavorable, e.g., when classifier retraining is slow [34], if several annotators are available in parallel [27], or with changeover costs of experiments and simulations [12]. In such cases, it seems natural to ask the oracle to annotate a *batch* of observations at a time. However, the benefit of batch active learning hinges on trade-offs between the cost of active learning, which comprises batch selection, classifier retraining, and the annotation effort, and the increase in classification accuracy. Literature tends to oversimplify these trade-offs: *Large batches yield slower learning rates, but they reduce the overall cost of active learning because of less frequent classifier training*, see [21, 23] for example. In reality, however, batch selection is more involved. For example it is difficult to model the cost terms or to select a good batch size. Further, selecting a good batch requires (i) quantifying the expected utility of a batch, and (ii) a strategy to traverse the space of candidate batches, which is prohibitively large. These issues have not been studied for one-class active learning so far. So this article is first to study when exactly batch one-class active learning is useful, compared to the sequential case. We break this general motivation down into three specific questions:

(Q1) How can batch utility be quantified in the one-class setting?

(Q2) What are suitable schemes to select candidate batches?

(Q3) Is there a sweet spot between the costs of batch active learning and classification accuracy?

Studying these questions is difficult. First, there are multiple ways to approach batch selection, with varying levels of complexity. One example are the different models for annotation costs. The costs can either be constant per label, can decrease with the batch size or may depend on the individual queries of the batch [27]. While some of these aspects have been mentioned in literature, there neither exists an overview nor a formal model of them. This makes it very hard to understand the trade-offs, to detect simplifications of the batch-selection problem, and to justify the benefit of batches.

Second, literature does not feature a method to calculate the batch utility for one-class active learning. In other domains, such as multi-class classification, the naive alternative to select the top-k sequential queries for a batch often is sub-optimal, since these observations tend to be similar to each other [26, 28, 29]. Different ways to approach this issue are conceivable, e.g., by introducing notions like batch diversity and

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

representativeness. But which notions actually are useful in a one-class setting, and how to possibly combine them into one aggregate measure is unclear.

Finally, batch selection is a combinatorial problem: there are $2^N - 1$ potential batches for N unlabeled observations. Current work on batch utility is confined to balanced domains and multi-class classification [9, 24, 38]. However, one-class problems have several distinctive properties, such as a significant class imbalance and undefined densities for the outlier class. These properties require specific sequential selection strategies [33], and we expect this for batch strategies as well.

Contributions. This article features the first principled approach to batch active learning for one-class classification. We make two specific contributions: (i) We formalize batch selection as a general and comprehensive optimization problem. Our framework trades off batch utility against annotation, batch-selection and classification costs under varying batch sizes. To our knowledge, this is the first formal framework that makes assumptions and simplifications for batch selection explicit. (ii) We propose several strategies to measure batch utility, specific to one-class classification. We then combine our utility measures with search strategies from four categories: top-k, iterative, partitioning and filtering strategies. We discuss their theoretical properties and compare them empirically on real-world data, against sequential selection and random baselines.

An important takeaway from this article is that batch queries with outlier detection are indeed different from their multi-class counterparts. In contrast to the multi-class case, batch diversity is not essential. Instead, selecting the top-k observations by informativeness suffices for good learning rates. There also is a sweet spot, and it is to use the top-k observations ranked by a sequential strategy with batch sizes between 8 and 16 observations. This decreases computational cost by up to an order of magnitude, while retaining the classification accuracy from the sequential case.

2 Formalization

In this section, we derive a formal model of batch selection, by framing it as an optimization problem. We start with some theoretical considerations and then discuss assumptions and simplifications.

Batch active learning selects a sequence of sets of observations for annotation and retrains a classifier after each set has been annotated. The ultimate objective is that the model has good classification accuracy after the last batch has been annotated. In general, annotations have a positive expected marginal utility. This is because each one provides additional information that helps to improve the classification model. But annotations may introduce a short-term bias, which can deteriorate classification accuracy temporarily [2]. This bias diminishes when more annotations become available. Next, one further assumes that active learning is subject to some budget restriction. For one, annotations require resources of real entities, such as humans or technical equipment. Next, retraining a classifier and searching for a good batch requires computational resources. One can assign resources specific costs, be they monetary, be they in time equivalents, and restrict the resources to the budget available. Existing cost-sensitive approaches merely focus on the annotation costs [8, 16, 31, 35] and lack a comprehensive model.

2.1 A Theoretical Model

Let $\mathcal{X}$ be a data set with N observations and M dimensions. The ground truth labels are $y \in \{\text{inlier}, \text{outlier}\}$. Formally, the objective of batch selection is to find a sequence of batches $\mathbf{B} = (B_1, \ldots, B_l)$ where $B_i \in \mathcal{P}(\mathcal{X}), B_i \cap B_{j \neq i} = \emptyset$ that yields the best possible classification accuracy *acc* on data $\mathcal{X}$ with labels y, with a semi-supervised classifier that trains a *model* on the annotated observations $\bigcup_{B_i \in \mathbf{B}} B_i = \mathcal{L}$ and the remaining observations $\mathcal{U}$, a budget T, and a cost function $c : B_i \mapsto \mathbb{R}$.

$$\begin{aligned} \underset{\mathbf{B}}{\text{maximize}} \quad & \text{acc}\left(\text{model}\left(\bigcup_{B_i \in \mathbf{B}} B_i \cup \mathcal{U}\right), \mathcal{X}, y\right) \\ \text{subject to} \quad & c(\mathbf{B}) \leq T \end{aligned}$$

This optimization problem only is of theoretical value, for two reasons. First, calculating *acc* requires access to ground truth labels. Since the very objective of active learning is to obtain these labels, one cannot solve this optimization problem directly. Instead, one must estimate the value of annotating a batch by a utility function u that is independent of a ground truth. One can interpret u as a proxy, i.e., a high utility value of a batch is expected to improve classification accuracy. An important property of u is that it depends on all batches that have already been annotated. This is because annotated batches provide information on the classification problem that the batch selection method in turn can use when selecting future batches. So the optimal sequence of batches is the one with the maximum cumulative utility, i.e., $\sum_{i=1}^{l} u(B_i | \bigcup_{j=1}^{i-1} B_j)$.

Second, the cost function is difficult to estimate. On the one hand, the different costs depend on each other. For instance, humans may idle during classifier retraining, but this only is cost-relevant when they cannot resort to some intermediary tasks. On the other hand, annotation costs may vary over time, e.g., because of the complexity of the query [27, 35] or because humans become more experienced with annotating.

While this optimization problem is theoretical, it is important to state it, to specify the goal of batch active learning. The problem is a basis for deriving another problem that one can solve in practice. Existing literature omits this step and does batch selection for each iteration of active learning independently.

2.2 A Relaxed Model A simplification is to relax the budget constraint, as follows. First, one only restricts the *number* of annotations. So the constraint becomes $|\bigcup_{i=1}^{l} B_i| \leq T,\ T \in \mathbb{N}$. The second simplification is to include the cost as independent and commensurable terms c_s (select), c_a (annotate) and c_t (train) in the objective function. This may require to normalize cost and utility terms. This gives

$$\begin{aligned} &\underset{\mathbf{B}}{\text{maximize}} && \sum_{i=1}^{l} u(B_i | \bigcup_{j=1}^{i-1} B_j) - c_s(B_i) - c_a(B_i) - c_t(B_i) \\ &\text{subject to} && |\bigcup_{i=1}^{l} B_i| \leq T, \quad T \in \mathbb{N}, \quad B_0 = \emptyset. \end{aligned}$$

Obtaining a solution to this relaxed problem still is difficult. Namely, as explained earlier, the utility of B_i depends on all previous batches. But the actual annotation of these batches is not available at the time of optimization. At first sight, a remedy would be to calculate the expected utility based on simulating all possible outcomes (inlier or outlier), for all batches. However, the state space to consider is $\mathcal{P}(\mathcal{X} \times \{\mathit{inlier}, \mathit{outlier}\})^l$. Traversing this space is prohibitive for a reasonably large number of observations, since it requires to retrain the classifier in each state.

2.3 A Practical Model To make solving the optimization problem feasible, one can use an additional simplification. Instead of optimizing for all batches $(B_i, \ldots, B_l)$, the optimization variable only is the current batch B_i, after $(B_1, \ldots, B_{i-1})$ have already been annotated. This allows to separate the problem into two nested problems. The inner one is to find a good batch for a given batch size k. For this inner problem, only annotation costs are relevant. The reason is that classifier retraining as well as the search for a batch are the same for all batch candidates of size k. The outer problem is to find an optimal k. Formally:

$$\max_k \left[\left(\underbrace{\max_{B_i \in \mathcal{P}_{=k}(\mathcal{U})} u(B_i | \bigcup_{j=1}^{i-1} B_j) - \sum_{x \in B_i} c_a(x, B_i)}_{\text{inner problem}} \right) - c_t(k) - c_s \cdot \binom{|\mathcal{U}|}{k} \right] \cdot \frac{T}{k}.$$

Here, c_t depends on k if the classifier is incremental, i.e., re-trained k times. Further, there are $\binom{|\mathcal{U}|}{k}$ possible batches of size k to consider. The last term $\frac{T}{k}$ is a technical constraint to adhere to the budget restriction. It implicitly assumes that k is fix for all batches, and that $k \in \{j \in \mathbb{N} | (\exists i \in \mathbb{N})[i \cdot j = T]\}$. However, this restriction is not crucial. In practice, one can repeat the optimization after the first batch is annotated, with the remaining budget $T' = T - k$. The nested problem now gives way to model the cost terms as follows.

Classifier training: There is an essential difference between incremental and non-incremental classifier training. If a classifier features dynamic updates, c_t linearly depends on k. Otherwise, c_t is constant, i.e., there only occurs one full retraining per batch. However, c_t is likely to be much higher for a full retraining than in the incremental case.

Batch Selection: c_s requires to calculate the utility for each possible batch. Thus, c_s is multiplied with the number of possible batches of size k.

Annotation: The costs of annotation are specific for each observation and may depend on the batch [27, 35]. Further, observations may be easier to annotate in sufficiently large batches [27]. However, a very common simplification is to assume identical annotation costs for all observations independently of the batch size [8]. In this case, $c_a(\cdot)$ reduces to $\bar{c}_a \cdot k$ and becomes part of the outer problem, since it only depends on k.

With these simplifications, the inner problem depends solely on u. This is convenient, since it allows to consider utility calculation and cost estimation independently. Put differently, the inner problem, i.e., finding a batch of size k with maximal utility, can be studied independently of any cost function. So the remaining problem to solve under the given simplifications is to find a batch of size k given already annotated observations $\mathcal{L}$ by evaluating a given utility function u. One can then vary k and u to find good batch sizes and a suitable utility function.

3 Method

In this section, we propose different batch strategies for one-class active learning. First, we discuss different criteria to measure *batch utility.* Then we present *batch strategies* to select a batch based on these criteria.

3.1 Batch Utility Intuitively, a utility function u quantifies the expected benefit of annotating one or more observations with respect to classification accuracy. In the sequential case, u is a function $u_{\text{seq}} : \mathcal{X} \rightarrow \mathbb{R}$, i.e., it assigns a utility per observation. For batch selection, the function is of type $u : \mathcal{P}(\mathcal{X}) \rightarrow \mathbb{R}$. So u quantifies the utility of a set. This allows to consider inter-observation relationships. For instance, annotating similar observations may have information overlap, and having them in one batch can be suboptimal. So u typically considers three criteria: informativeness, representativeness, and diversity [28]. In the following, we elaborate on these criteria, review different possibilities to implement them with one-class classifiers and describe our choice for this current article. We focus on general ways to quantify these criteria and do not consider application-specific ones, like remote sensing [30] or document relevance [39].

3.1.1 Informativeness Informativeness is a function $\tau(x)$ that quantifies how much a classifier is expected to benefit from knowing the label of a single observation $x \in \mathcal{X}$. There are several categories of informativeness functions, and many of them have been proposed explicitly for one-class classification [33]. In general, one can distinguish between data-based and model-based informativeness. Data-based informativeness solely depends on $\mathcal{L}$ and $\mathcal{U}$. An example is to attribute high informativeness to observations where the distribution of the class probability has a high entropy [13]. Model-based informativeness depends on a trained classifier. One example is the decision-boundary strategy, where observations are more informative the closer they are to the decision boundary. There also are hybrid strategies that combine both aspects [15, 40].

Our choice: Based on the experimental evaluation in [33], we choose two model-based functions. Let $f(x)$ be the decision function of a one-class classifier, which returns $f(x_i) > 0$ for outliers and ≤ 0 for inliers. The first function is the decision-boundary strategy $\tau_{\text{DB}} = -|f(x)|$, which gives high informativeness to regions of high classification uncertainty. The second function is $\tau_{\text{HC}} = f(x)$, which prefers observations which are predicted to be an outlier with high confidence [1].

3.1.2 Representativeness Representativeness is a function $rep(x)$ that quantifies how well an observation represents the underlying data set, and observations doing this well are preferred. However, there are different interpretations of representativeness. An observation can be representative if it lies in a dense area of the data distribution. This can be quantified by kernel density estimation [18] or by calculating the average distance to the k-nearest neighbors [20, 41]. A different, implicit, approach is to cluster data and select the cluster medoids as representative observations [10, 29, 39]. Finally, some approaches quantify representativeness for a batch [7, 9, 11, 37, 38]. They use maximum mean discrepancy (MMD) to measure how well the batch follows the full data distribution.

Literature has proposed strategies that use a combined notion of representativeness and informativeness for sequential query selection, e.g., a linear combination of the distance to the decision boundary and to the nearest neighbors [40]. Whenever such a sequential strategy is applicable in the following, we make this clear by writing u_{seq} instead of τ or *rep*.

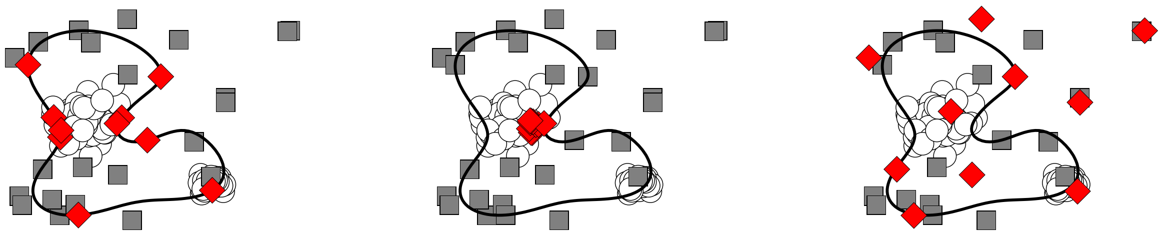
Our choice: In our article, we quantify representativeness using kernel density estimation. However, we also present two clustering approaches that choose representative queries implicitly, see Section 3.2.3.

3.1.3 Diversity While the first two criteria are defined for individual observations, the diversity criterion is for batches. Intuitively, a batch is diverse if its observations are dissimilar to each other. High diversity of a batch is good, since dissimilar observations are expected to have little information overlap. There are several ways to enforce diversity either implicitly or explicitly. One explicit method is to maximize pair-wise distances, e.g., in the data or kernel space [3, 7]. Cluster-based strategies consider diversity implicitly, since they select observations from different clusters, which is likely to yield a diverse batch.

Our choice: We follow previous work to quantify the diversity div of a batch B as the minimum pair-wise distance d of two observations $div(B) = \min_{x_i, x_j \in B} d(x_i, x_j)$ [3, 6, 20]. We use two distance functions. The first one is the Euclidean distance d_{ED} in the data space. The second one is a distance in the reproducing kernel Hilbert space of a kernel-based classifier $d_{\text{AK}} = -|\cos(\angle(\phi(x_i), \phi(x_j)))|$. Intuitively, d_{AK} is proportional to the angle of two observations in the kernel space [3].

A key challenge is to combine these so-called batch criteria [28] into a single utility measure. Before addressing this, we discuss some theoretical properties of the criteria in the context of one-class classification for outlier detection. First, with outlier detection, a large share of the data space is sparse. Given such a sparsity, diversity may not be useful. This is because it is likely to include observations from sparse regions, which only provide little information for the classifier. Second, with outlier detection, representativeness may not be very useful either. The reason is that it focuses on regions of high density, which are likely to contain only observations from the inlier class. The following example illustrates these properties.

Example 3.1. *Figure 1 shows a 2-dim synthetic data set with inliers (white circles) and outliers (gray squares). We fit a one-class classifier, considering all*



(a) Informativeness (b) Representativeness (c) Diversity

Figure 1: Comparison of batch selection based on different criteria.

observations as unlabeled; the decision boundary is the black line. We then compute batches of 10 observations for each criterion independently (red diamonds).

As expected, representativeness selects the observations where the data density is high. The resulting batch contains a bulk of observations in a small region, with high similarity. With diversity, the observations in the batch are well spread. Yet some queries lie in very sparse regions where feedback might only influence the classification of very few observations. Informativeness with τ_{DB} selects the 10 observations closest to the decision boundary, see Figure 1a. Although the batch does not cover the full data space, the visual inspection shows that the observations selected are diverse and representative, without explicit consideration of these criteria.

Given this, we propose the following hypothesis.

Hypothesis 3.1. (Batch Criteria) *With one-class outlier detection, representativeness and diversity criteria are not useful for batch selection.*

Our experiments on real world data will confirm this hypothesis. The hypothesis also holds for weighted combinations of the criteria, see Section 4.

3.2 Batch strategies In general, many ways to combine batch criteria to select a batch are conceivable. Some strategies have been proposed for binary and multi-class settings [9, 38]. We only are aware of one summary for multi-label data [24] which requires other approaches than our one-class setting. Further, there does not seem to be any strategy specific to one-class classification. It is unclear whether existing multi-class approaches transfer to the one-class setting. In this section, we therefore propose different batch selection strategies for one-class active learning. We classify them into four categories: baseline, iterative, partitioning, and filtering strategies.

3.2.1 Baselines As one baseline, we propose *random batch*, which samples k unlabeled observations independent of any criteria.

Algorithm 1: Iterative Strategy

```
Input  : $\tau(x)$, $rep(x)$, $div(B)$, $\lambda_{\text{inf}}, \lambda_{\text{rep}}, \lambda_{\text{div}}$, $k$
Output: $B$
$B = \left\{ \arg\max_{x_i \in \mathcal{U}} \left( \lambda_{inf} \cdot \tau(x_i) + \lambda_{rep} \cdot rep(x_i) \right) \right\}$
for $i \leftarrow 2 \ldots k$ do
    $x_i^* = \arg\max_{x_i \in \mathcal{U} \setminus B} \left( \lambda_{\text{inf}} \cdot \tau(x_i) + \lambda_{\text{rep}} \cdot rep(x_i) + \lambda_{\text{div}} \cdot div(B \cup \{x_i\}) \right)$
    $B = B \cup \{x_i^*\}$
end
```

The second baseline is *TopK*, a straightforward extension of the sequential mode. The idea is to calculate utility for each observation independently, with a given u_{seq}, and then choose the top k observations.

3.2.2 Iterative Strategy *Iterative* strategies are heuristics to find a batch that maximizes the weighted sum of batch criteria. Formally, this is

$$u(B) = \sum_{x \in \mathcal{B}} \lambda_{\text{inf}} \cdot \tau(x) + \lambda_{\text{rep}} \cdot rep(x) + \lambda_{\text{div}} \cdot div(B), \quad (3.1)$$

with weight parameters $\lambda_{\text{inf}}, \lambda_{\text{rep}}, \lambda_{\text{div}} \in \mathbb{R}$ to specify the importance of the three criteria. A fundamental difficulty with this approach is that one must calculate u for $\binom{|\mathcal{U}|}{k}$ candidate batches. So to find a good solution, one can instead build the batch in a greedy way [17, 39]. This is, the observation that maximizes the weighted sum of the three criteria is added to the batch in each iteration until the batch contains k observations; see Algorithm 1. If τ, *rep* and *div* are submodular, the greedy solution has a lower bound of $(1 - \frac{1}{e}) \cdot u(B^*)$, relative to the optimal utility $u(B^*)$ [17].

3.2.3 Partitioning Strategies A different approach is to put emphasis on diversity. The idea is to first divide the data set into k random or disjoint subsets and then

Algorithm 2: Cluster-TopK Strategy

```
Input  : $\tau(x)$, $k$, $m \geq k$
Output: $B$
$\mathcal{M}$ = top $m$ observations from $\mathcal{U}$ ranked by $\tau(x)$
$\mathcal{C} = KMedoidsClustering(\mathcal{M}, k)$
$B = \emptyset$
foreach cluster $C \leftarrow \mathcal{C}$ do
    $x_C$ = medoid of $C$
    $B = B \cup \{x_C\}$
end
```

Algorithm 3: Ensemble Strategy

```
Input  : $\mathcal{X}_1, \dots, \mathcal{X}_k$, $C_1, \dots, C_k$, $u_{\text{seq}}$
Output: $B$
$B = \emptyset$
foreach sample $\mathcal{X}_i$ with classifier $C_i$ do
    $x_i^* = \underset{x_i \in \mathcal{X}_i, x_i \notin B}{\arg\max}\ u_{\text{seq}}(x_i)$
    $B = B \cup \{x_i^*\}$
end
```

Algorithm 4: Filter Similar Strategy

```
Input  : $\tau(x)$, $d(x_i, x_j)$, $k$
Output: $B$
$B = \mathcal{U}$
while $|B| > k$ do
    $a, b = \underset{x_i, x_j \in B}{\arg\min}\ d(x_i, x_j)$
    $B = B \setminus \{\underset{x \in \{a,b\}}{\arg\min}\, \tau(x)\}$
end
```

select one observation from each subset. To this end, two common approaches are to *cluster* data [10, 22, 28, 29, 39] and to train several classifiers on different samples of the data as an *ensemble* [22].

One *cluster* strategy partitions $\mathcal{U}$ into k clusters, for instance by k-medoids clustering. The medoids of each cluster then form the batch. *Cluster* only considers representativeness and diversity, but not informativeness.

An extension is *cluster-TopK*, see Algorithm 2. The idea is to include informativeness to filter unlabeled observations. This is, *cluster-TopK* selects the m highest ranked unlabeled observations according to $\tau(x)$ where $|\mathcal{U}| \gg m \gg k$ and clusters them into k clusters. As before, the medoids of each cluster are the batch B.

An *ensemble* strategy randomly samples k subsets of size $|\mathcal{X}|/k$, see Algorithm 3. For each subset, the strategy trains a classifier and selects the best observation according to a given u_{seq}.

Algorithm 5: Filter Hierarchical Strategy

```
Input  : $\tau(x)$, $rep(x)$, $div(B)$, $k$
Output: $B$
$\mathcal{M} = 4 \cdot k$ highest ranked $x$ in $\mathcal{U}$ by $rep(x)$
$\mathcal{M} = 2 \cdot k$ highest ranked $x$ in $\mathcal{M}$ by $\tau(x)$
$B$ = {select highest ranked $x_i$ in $\mathcal{M}$ by $\tau(x)$}
for $i \leftarrow 2 \dots k$ do
    $x_i^* = \underset{x_i \in \mathcal{M} \setminus B}{\arg\max}\, div(B \cup \{x_i\})$
    $B = B \cup \{x_i^*\}$
end
```

3.2.4 Filtering Strategies Finally, there are filter strategies which proceed top-down to select a batch. The idea is to start with all unlabeled observations and step-wise apply filter criteria until only k observations are left. We see two types of *filter* strategies.

The *filter similar* strategy searches for the two most similar observations and removes the one with less informativeness [25], see Algorithm 4.

The *filter hierarchical* strategy filters for each of the three criteria, step by step [19], see Algorithm 5. This is, *filter hierarchical* first selects the $4 \cdot k$ most representative observations and from these the $2 \cdot k$ most informative ones. From them, it greedily selects the batch, similarly to the iterative strategy.

All batch strategies presented have the same objective, increasing classification accuracy. But the realizations differ significantly. Based on plausibility arguments, there is no single, superior approach. While literature has proposed strategies for the multi-class setting, it is unclear which conclusions transfer to the one-class setting. We now derive two hypotheses. If they hold, they help with the selection of a suitable approach in the one-class setting.

The first hypothesis extends Hypothesis 3.1 to batch strategies.

Hypothesis 3.2. (Batch Selection Strategy) *With one-class outlier detection, a strategy solely based on informativeness is expected to outperform more sophisticated strategies that explicitly incorporate representativeness and diversity.*

Based on this hypothesis, we expect *TopK* to perform well, compared to more sophisticated approaches.

The second hypothesis concerns the cost trade-offs, see Section 2. Batch selection strategies have different complexity. When assuming constant evaluation costs for u_{seq} and τ, we derive the following complexities: constant for *random* batches, $\mathcal{O}(n)$ for *TopK*, $\mathcal{O}(kn)$ for *Iterative*, approximately $\mathcal{O}(kndi)$ with dimensionality

d and a number i of clustering steps for *cluster-based*, $\mathcal{O}(k\,(n/k)^2)$ for *Ensemble* where k classifiers are learned on n/k large partitions, $\mathcal{O}(k+n^2)$ for *Filter Similar* where n^2 is the complexity of computing the similarity matrix, and $\mathcal{O}(kn)$ for *Filter Hierarchical*. With this, we expect the runtime cost of the batch selection strategies to be low compared to the classifier training, which requires solving a quadratic optimization problem for standard one-class classifiers, like SVDDneg. All this motivates the following hypothesis.

HYPOTHESIS 3.3. (COST TRADE-OFFS) *The classifier cost c_t dominates the batch selection costs c_s during active learning with batches in most cases.*

When this hypothesis holds, batch selection costs c_s are not relevant, and we can simplify the optimization model, see Section 2.3. An exception is *Ensemble*, which requires to train k classifiers on n/k partitions of the data set. Our experiments on real world data confirm both hypotheses, see Section 4.

4 Experiments

We now present our empirical findings and discuss them in the context of the three hypotheses introduced earlier. Our implementation, raw results, and notebooks are available at `https://www.ipd.kit.edu/bocal`.

4.1 Setup

We use 21 standard data sets for outlier detection [5]. Each data set comes in three resamples, with an outlier ratio of 5%, and up to 1000 observations.

Classifier: Our base classifier is SVDDneg [32] with the Gaussian kernel, tuned as proposed in [36], and the cost parameter set as proposed in [32]. We also use the obtained kernel parameter for kernel density estimation and for the kernel angle distance d_{AK}.

Active Learning: We choose τ_{DB} as the informativeness criterion, since it has yielded the best results in preliminary experiments of ours. For diversity, we use d_{AK} unless stated differently. We set $m = 10k$ for *Cluster-TopK*. We start with no labels, a budget of $T = 128$ and evaluate $k \in \{1, 2, 4, 8, 16, 32, 64, 128\}$.

Evaluation Metrics: We evaluate classification accuracy with the Matthews Correlation Coefficient (MCC), which is well suited for imbalanced data. MCC returns values in $[-1, 1]$; higher values are better. We report the end quality (EQ), i.e., the accuracy after the budget is exhausted, as the median over all data sets. Our experiments run on an AMD Ryzen Threadripper 2990WX with 64 virtual cores. We measure the total runtime t and the query selection time t_s in seconds.

4.2 Results

We first evaluate the usefulness of the three batch criteria in the one-class setting. We then compare the batch strategies presented in Section 3. We compare all strategies against the sequential mode and evaluate the sensitivity to varying batch sizes. Finally, we present runtime measurements and discuss them with resepect to the trade-offs introduced in Section 2.

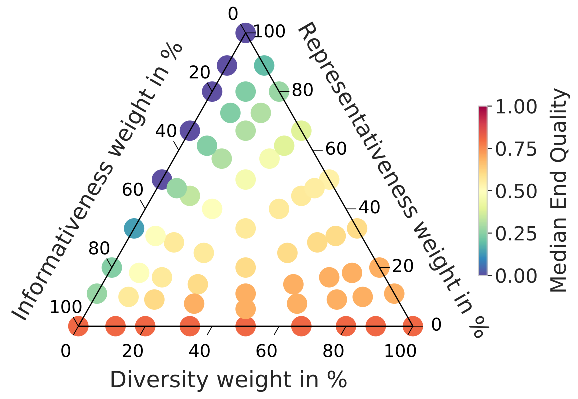


Figure 2: Influence of batch criteria on end quality.

Batch Criteria We evaluate the usefulness of batch criteria by varying the weights of the *iterative* strategy. Figure 2 shows the median EQ for the different combinations with a batch size of $k = 4$. The EQ is highest for $\lambda_{\text{rep}} = 0$, as indicated by the red dots on the bottom line of the triangle. Results with different batch sizes or τ_{HC} as the informativeness criterion are similar. With d_{ED} as diversity, a small value $\lambda_{\text{rep}} > 0$ does not reduce EQ as much as for d_{AK} but $\lambda_{\text{rep}} = 0$ also results in the highest EQ. On a per-data-set level, there are some instances where setting $\lambda_{\text{div}} > 0$ has a positive effect on EQ. However, tuning the weight parameters per data set is unrealistic. Namely, this would require a labeled training set. We conclude that a combination of the three criteria with strictly positive weights does not increase the model quality in the one-class setting. Informativeness has a dominating influence on EQ – this supports Hypothesis 3.1. So $\lambda_{\text{inf}} = 1, \lambda_{\text{rep}} = 0, \lambda_{\text{div}} = 0$ is the best choice, which corresponds to the *TopK* strategy. This is a strong difference to the balanced and multi-class domain [20].

Batch Strategies Next, we compare the performance of the different batch strategies proposed in this article. See Table 1 for the median EQ. *TopK* outperforms all other strategies. Up to a batch size of $k = 8$, the EQ is 0.81 and hence equal to the sequential strategy. Up to $k = 64$, the accuracy loss is small compared to the sequential strategy. The more complex partitioning or filtering strategies generally yield results similar to or worse than *TopK*. This supports Hypothesis 3.2. These results are again contrary to multi-class batch strategies, where partitioning and iterative strate-

k	rand-B	TopK	Cluster	ClusterTopK	Ensemble	FilterSim	FilterHrch	Sequential
1	0.49	**0.81**	0.00	0.70	**0.81**	0.70	0.00	**0.81**
2	0.43	**0.81**	0.28	**0.81**	**0.81**	**0.81**	0.64	-
4	0.49	**0.81**	0.35	0.79	**0.81**	0.79	0.78	-
8	0.49	**0.81**	0.54	0.76	**0.81**	0.78	0.79	-
16	0.49	**0.80**	0.57	0.68	**0.80**	0.78	0.77	-
32	0.49	**0.79**	0.60	0.62	†	0.75	0.75	-
64	0.49	**0.79**	0.57	0.60	†	0.70	0.74	-
128	0.49	**0.75**	0.57	0.57	†	0.70	0.71	-
Mean Rank‡	5.10	**1.99**	4.72	3.42	-	2.79	2.98	-

Table 1: Median EQ over all data sets in comparison to a sequential baseline. †Optimization problem infeasible for $k > 16$. ‡ Rank calculated for each data set and batch size. Sequential and ensemble strategy are excluded.

k	rand-B	TopK	Cluster	ClusterTopK	Ensemble	FilterSim	FilterHrch	Sequential
1	129s/0.1%	158s/6.7%	179s/1.9%	213s/6.3%	344s/49.2%	180s/12.4%	158s/13.4%	178s/5.9%
2	70s/0.1%	71s/6.2%	99s/1.8%	79s/5.4%	117s/34.7%	99s/10.3%	93s/7.5%	-
4	41s/0.0%	45s/5.3%	58s/0.3%	44s/5.9%	53s/34.2%	55s/11.0%	40s/9.1%	-
8	18s/0.1%	24s/5.1%	23s/0.1%	19s/7.1%	35s/45.9%	22s/11.0%	23s/7.4%	-
16	10s/0.0%	12s/5.0%	13s/0.2%	14s/5.8%	25s/59.6%	14s/10.4%	12s/6.6%	-
32	6s/0.0%	5s/5.4%	7s/0.1%	6s/6.1%	-	8s/13.4%	6s/8.8%	-
64	3s/0.0%	4s/3.5%	3s/0.3%	4s/4.7%	-	4s/11.0%	4s/5.6%	-
128	2s/0.0%	2s/3.6%	3s/0.4%	3s/2.9%	-	3s/7.8%	3s/6.4%	-

Table 2: Median experiment run time t in seconds and ratio t/t_{s} of time spent for query selection in %.

gies outperform *TopK* [29, 39], and where end quality increases with the batch size in some cases [24].

Trade-offs Table 2 shows the t and t_{s} for varying batch sizes. In all cases, t decreases considerably with increasing batch size. As expected, batch selection makes up only a fraction of the overall runtime, except for *ensemble*. The total experiment run time is roughly proportional to $1/k$. The dominating factor is the number of classifier trainings T/k.

Overall, the runtime costs of classifier training are two magnitudes higher than the ones of batch selection, i.e., $c_t \gg c_s$. This confirms Hypothesis 3.3. We conclude that, in a one-class setting, computational costs can decrease by a factor of 10 with *TopK* without sacrificing accuracy, compared to the sequential case. More sophisticated batch selection strategies do not improve results in the one-class setting (Q2). So the sweet spot between active learning costs and classification accuracy is to use *TopK* batches with decision boundary informativeness (Q1) and setting k to a value in $[8, 16]$ (Q3). The sweet spot relies on the assumption that annotation costs are fix, see Section 2. However, our conclusions also hold if annotation costs decrease with the batch size until $k = 8$. Namely, increasing the batch size does not affect classification accuracy.

5 Conclusions

Batch active learning gives way to annotating observations in parallel when classifier retraining is slow, or experimental changeover costs are high. To utilize computational and annotation resources most efficiently, we strive to find a sweet spot between the costs of one-class batch active learning and the improvement in classification accuracy. To this end, we present a formal framework for batch query selection. Based on it, we propose several batch selection methods tailored towards one-class classification. Our general considerations and experiments show that selecting the top-k observations according to a sequential query strategy is a dominant choice, compared to more sophisticated strategies. This is a finding which is different from the situations in multi-class and binary active learning. Batch active learning even achieves the classification accuracy of a sequential strategy, while reducing computational costs by an order of magnitude.

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] V. Barnabé-Lortie, C. Bellinger, and N. Japkowicz. "Active Learning for One-Class Classification". In: *ICMLA*. 2015.

[2] J. Bernard et al. "VIAL: a unified process for visual interactive labeling". In: *The Visual Comp* 34.9 (2018).

[3] K. Brinker. "Incorporating Diversity in Active Learning with Support Vector Machines". In: *ICML*. 2003.

[4] M. Bunse, A. Saadallah, and K. Morik. "Towards Active Simulation Data Mining". In: *ECML PKDD Workshop*. 2019.

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

[6] T. Cardoso et al. "Ranked batch-mode active learning". In: *Information Sciences* 379 (2017).

[7] S. Chakraborty, V. Balasubramanian, and S. Panchanathan. "Adaptive Batch Mode Active Learning". In: *IEEE Trans Neural Netw Learn Syst* 26.8 (2015).

[8] S. Chakraborty, V. Balasubramanian, and S. Panchanathan. "An optimization based framework for dynamic batch mode active learning". In: *NIPS* (2010).

[9] R. Chattopadhyay et al. "Batch Mode Active Sampling Based on Marginal Probability Distribution Matching". In: *TKDD* 7.3 (2013).

[10] B. Demir, C. Persello, and L. Bruzzone. "Batch-Mode Active-Learning Methods for the Interactive Classification of Remote Sensing Images". In: *IEEE Trans Geosci Remote Sens* 49.3 (2010).

[11] B. Du et al. "Exploring Representativeness and Informativeness for Active Learning". In: *IEEE Trans Cybern* 47.1 (2017).

[12] R. Ghani and M. Kumar. "Interactive learning for efficiently detecting errors in insurance claims". In: *SIGKDD*. 2011.

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

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

[15] N. Görnitz et al. "Toward Supervised Anomaly Detection". In: *JAIR* (2013).

[16] R. Haertel et al. "Return on investment for active learning". In: *NIPS Workshop*. 2008.

[17] S. Hoi et al. "Batch mode active learning and its application to medical image classification". In: *ICML*. 2006.

[18] T. Hospedales, S. Gong, and T. Xiang. "Finding rare classes: Active learning with generative and discriminative models". In: *TKDE* 25.2 (2011).

[19] Y. Jiao et al. "A Multicriterion Query-Based Batch Mode Active Learning Technique". In: *Foundations of Intelligent Systems*. 2014.

[20] S. Kee, E. del Castillo, and G. Runger. "Query-by-committee improvement with diversity and density in batch active learning". In: *Inf Sciences* 454 (2018).

[21] D. Lewis and W. Gale. "A Sequential Algorithm for Training Text Classifiers". In: *SIGIR*. 1994.

[22] X. Li, R. Guo, and J. Cheng. "Incorporating Incremental and Active Learning for Scene Classification". In: *ICMLA*. 2012.

[23] J. O'Neill, S. Jane Delany, and B. MacNamee. "Model-Free and Model-Based Active Learning for Regression". In: *AISC*. Vol. 513. 2017.

[24] O. Reyes and S. Ventura. "Evolutionary strategy to perform batch-mode active learning on multi-label data". In: *TIST* 9.4 (2018).

[25] D. Sadigh et al. "Active preference-based learning of reward functions". In: *Robotics: Science & Syst.* 2017.

[26] B. Settles. *Active learning*. Tech. rep. 2012.

[27] B. Settles. "From theories to queries: Active learning in practice". In: *AISTATS Workshop*. 2011.

[28] D. Shen et al. "Multi-criteria-based active learning for named entity recognition". In: *ACL*. 2004.

[29] X. Shen and C. Zhai. "Active feedback in ad hoc information retrieval". In: *SIGIR*. 2005.

[30] Q. e. a. Shi. "Spatial coherence-based batch-mode active learning for remote sensing image classification". In: *IEEE Trans Image Process* 24.7 (2015).

[31] Y.-P. Tang and S.-J. Huang. "Self-paced active learning: Query the right thing at the right time". In: *AAAI*. 2019.

[32] R. Tax David and Duin. "Support Vector Data Description". In: *Machine Learning* (2004).

[33] H. Trittenbach, A. Englhardt, and K. Böhm. "An Overview and a Benchmark of Active Learning for Outlier Detection with One-Class Classifiers". In: *arXiv:1808.04759* (2018).

[34] H. Trittenbach, A. Englhardt, and K. Böhm. "Validating One-Class Active Learning with User Studies–a Prototype and Open Challenges". In: *ECML PKDD Workshop*. 2019, p. 17.

[35] Y.-L. Tsou and H.-T. Lin. "Annotation cost-sensitive active learning by tree sampling". In: *Machine Learning* 108.5 (2019).

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

[37] Z. Wang et al. "A batch-mode active learning framework by querying discriminative and representative samples for hyperspectral image classification". In: *Neurocomputing* 179 (2016).

[38] Z. Wang and J. Ye. "Querying discriminative and representative samples for batch mode active learning". In: *TKDD* 9.3 (2015), p. 17.

[39] Z. Xu, R. Akella, and Y. Zhang. "Incorporating Diversity and Density in Active Learning for Relevance Feedback". In: *Advances in Information Retrieval*. Vol. 4425. 2007.

[40] L. Yin, H. Wang, and W. Fan. "Active learning based support vector data description method for robust novelty detection". In: *Knowl-Based Syst* 153 (2018).

[41] X. Zhu, J. Lafferty, and Z. Ghahramani. "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions". In: *ICML Workshop*. 2003.