Systematic Evaluation of CASH Search Strategies for Unsupervised Anomaly Detection

Ioannis Antoniadis IOANNIS.ANTONIADIS@STUDENT.KULEUVEN.BE
Vincent Vercruyssen VINCENT.VERCRUYSSEN@KULEUVEN.BE
Jesse Davis JESSE.DAVIS@KULEUVEN.BE
Dept. of Computer Science, Celestijnenlaan 200A, 3001 Leuven, Belgium.

Editor: Nuno Moniz, Paula Branco, Luís Torgo, Nathalie Japkowicz, Michał Woźniak and Shuo Wang.

Abstract

Anomaly detection is an important data mining task that aims to detect abnormal examples in a dataset. Dozens of unsupervised algorithms have been developed for this task, each of which can be finely controlled via multiple hyperparameters. Therefore, choosing an algorithm that works well for a new dataset has traditionally been a time-consuming trial-and-error process. Moreover, any ground-truth labels to guide this process are hard to come by in real-world anomaly detection problems. On the other hand, if we are able to collect a small, labeled validation set, we could leverage the AutoML paradigm to automate this model search. While the off-the-shelf AutoML search strategies for combined algorithm selection and hyperparameter optimization (CASH) are effective for supervised classification and regression tasks, they require the availability of plenty of ground-truth labels and large validation sets. It is unclear whether CASH will be equally effective for anomaly detection problems where the validation sets are typically small at best and not always representative of the test set at worst. In this paper, we present a discussion and experimental evaluation of how the structure of the validation set, i.e., its size and label bias, impacts the performance of different CASH search strategies within the context of anomaly detection.

Keywords: anomaly detection, model selection, AutoML, CASH

1. Introduction

The goal of anomaly detection is to identify examples in a dataset that are somehow different or abnormal. Anomaly detection is important in practice as anomalies often correspond to costly or undesired behavior such as fraudulent credit card use (Pourhabibi et al., 2020), excess water consumption (Vercruyssen et al., 2018), or suspicious DNS traffic (Robberechts et al., 2018). Anomaly detection is an extremely challenging problem for several reasons. First, anomalies are by definition exceedingly rare, meaning that it is an imbalanced learning problem. Second, anomalies may not follow a particular pattern that a supervised learner could pick up on. Third, anomaly detection problems often lack (large amounts of) labeled data because it is costly to collect them. This is particularly true for anomalous behavior as we either have to wait until it occurs naturally (e.g., a machine failure) or purposely cause it to arise by, e.g,. breaking the system. The latter is often not feasible, as for example, no one will consent to allowing a machine to break just to collect some data.

While researchers have designed many anomaly detection algorithms (Liu et al., 2008; Breunig et al., 2000; Ramaswamy et al., 2000; Li et al., 2020c; He et al., 2003; Campos et al., 2016; Emmott et al., 2015), picking and configuring the one best suited for the task at hand remains difficult. Algorithms can focus on finding different types of anomalies and their detection performance can be optimized by tuning various hyperparameters. The lack of labels and the imbalanced nature of the problem means that practitioners often resort to selecting a model manually via trail-and-error, or they stick to one algorithm with its default hyperparameters (Domingues et al., 2018; Emmott et al., 2015).

More recently, Soenen et al. (2021) advocated for a more standard machine learning style approach of (1) collecting a small labeled *validation set*, (2) training different combinations of algorithm and hyperparameter settings, (3) measuring their performance on the validation set, and (4) selecting the best-performing model. However, that work relied on using a naive and computationally expensive grid search over all possible algorithm configurations. In contrast, one of the big successes of automated machine learning (AutoML) has been *combined algorithm selection and hyperparameter optimization* (CASH) (Hutter et al., 2019). In this area there has been significant progress in designing smart search strategies that can quickly identify the appropriate algorithm and its configuration for a given dataset.

The goal of this work is to examine the usefulness of CASH search strategies for selecting and configuring anomaly detectors. However, it is non-obvious whether the findings and approaches from the CASH strategies developed for supervised classification and regression, will transfer to the anomaly detection setting because:

- The costs associated with obtaining labels means that it is likely that any validation set one can obtain, will be exceedingly small.
- The rarity of anomalies means that they will likely be more prevalent in the validation set than in the wild to ensure that there is enough signal to compute an evaluation metric (in order to guide the model search).
- The fact that certain anomalies are easier to identify and hence label means that the observed labels are biased. That is, the validation set labels are not an i.i.d. sample.

We empirically evaluate the effectiveness of several guided and unguided CASH search strategies to select the best among five distinct anomaly detectors for six datasets. Furthermore, we consider the effect of how the size of the validation set and its label bias affect the performance of the different search strategies. The label bias captures whether the validation set is a representative sample of the full dataset or not. We found that: (1) even simple CASH search strategies already outperform naively sticking with an algorithm's default hyperparameters, (2) larger validation sets guide the search to models that generalize better to an unseen test set, (3) a biased validation set surprisingly allows the selection of better performing models than a validation set that is an i.i.d. sample of the full dataset, and (4) increasing the size of the search space (i.e., the number of possible algorithm-hyperparameter configurations) improves the performance of the CASH search strategies.

2. Preliminaries

Given a feature space $\mathcal{X}$ and a label space $\mathcal{Y}$, let $D = \{(x_i, y_i)\}_{i=1}^{n}$ be a dataset consisting of n tuples such that $x_i \in \mathcal{X}$ is an example and $y_i \in \mathcal{Y}$ its label. In anomaly detection, this label is binary indicating whether example x_i is *normal* or *anomalous*. Let $D_{train} = \{(x_i)\}_{i=1}^{l}$ be a training split consisting only of features $\mathcal{X}$ and $D_{valid} = \{(x_i, y_i)\}_{i=1}^{m}$ a validation split that also includes each example's label, such that $D_{train} = D \setminus D_{valid}$ and $1 \leq m, l \leq n$. The CASH problem can now be formally defined as follows (Hutter et al., 2019):

Problem 1 (CASH for anomaly detection):
Given D_{train}, D_{valid}, a set of k anomaly detectors $\mathcal{A} = \{A^1, ..., A^k\}$ with their respective hyperparameter domains $\Lambda^1, ..., \Lambda^k$, and a resource budget $\mathcal{T}$, find the anomaly detector A^* and a corresponding hyperparameter configuration λ^* that minimize some loss function $\mathcal{L}$ on D_{valid}:

$$A^*, \lambda^* \in \underset{A^j \in \mathcal{A},\, \lambda \in \Lambda^j}{\arg\min}\ \mathcal{L}(A^j_\lambda, D_{train}, D_{valid}) \tag{1}$$

under the constraint imposed by the resource budget $\mathcal{T}$.

In this paper, we use the *area under the receiver operating characteristic* (AUC) as the loss function[1] as this metric is commonly used in anomaly detection (Campos et al., 2016).

CASH solvers for Equation (1) need to overcome several difficulties such as the complexity and high-dimensionality of the combined *configuration search space*, the absence of gradients for the loss function, and various resource constraints (Luo, 2016). Such solvers are either model-free (unguided) or model-based (guided) (Hutter et al., 2019). The most common unguided strategies are *grid search* and *random search*. Grid search evaluates the loss function $\mathcal{L}$ for each possible configuration (A^k, λ^k) in the search space (Hutter et al., 2014). This strategy is clearly extremely inefficient, especially if the structure of the loss function is non-uniform over the full search space. Random search operates by randomly sampling configurations (A^k, λ^k) from the search space until the budget $\mathcal{T}$ is exhausted (Bergstra and Bengio, 2012). While superior to grid search, random search might be sub-optimal if the search space is conditional because it will favor configurations sampled from the larger sub-spaces. As an alternative to random search, we propose *uniform exploration* which works by dividing the total search budget $\mathcal{T}$ equally among the candidate sub-spaces and exploring them independently with a random search subroutine. This increases the probability of fairly exploring all the constituent sub-spaces regardless of their relative size.

Contrary to the model-free solvers, the guided solvers assume that the loss function $\mathcal{L}$ has a meaningful structure with relation to the search space and that this structure can be learned. The state-of-the-art *sequential model-based algorithm configuration* (Smac) framework uses *Bayesian optimization* to simultaneously learn this structure and exploit it to (approximately) find the best-performing configuration (A^*, λ^*) (Hutter et al., 2011; Lindauer et al., 2022). The framework uses a surrogate model (random forest) to learn the target function and iteratively updates it by observing additional values and by optimizing a cheap acquisition function (expected improvement) to select the next configuration to evaluate. Smac can handle the (categorical) hyperparameters of the anomaly detectors and the conditional parameters of the CASH search space, by default, while it scales well for large datasets and high-dimensional search spaces.

1. To properly utilize AUC as the loss function, we simply set $\mathcal{L} = 1 - \text{AUC}$.

3. Related Work

The idea of applying AutoML for anomaly detection has been explored in several works (Li et al., 2020a; Chakraborty et al., 2020; Lai et al., 2020; Li et al., 2020b; Zhao et al., 2020). However, *none* of these works question the assumed availability of a large, unbiased, fully-labeled validation set to guide the model search. In contrast, our work aims to evaluate the utility of CASH for anomaly detection in light of the limited label availability that characterizes the anomaly detection setting. We now give a brief overview of said works.

AutoOD Li et al. (2020a) proposed AutoOD to tackle the *neural architecture search* problem for unsupervised anomaly detection using deep autoencoders. AutoOD employs a LSTM-based meta-learning search strategy and its objective function is evaluated on a labeled validation set. No open-source implementation of AutoOD is currently available.

Luminaire Chakraborty et al. (2020) developed Luminaire as an automated anomaly detection system for time-series data. The search space is comprised of data preparation algorithms, such as imputation and smoothing, and time series modeling algorithms, such as structural and filter-based models. Artificially generated anomalies are injected at different scales to facilitate supervised training.

TODS Lai et al. (2020) designed TODS for time series outlier detection. TODS has five modules: (1) data processing, (2) time series processing, (3) feature analysis, (4) algorithm detection, and (5) reinforcement learning. The latter module provides the capability to inject human knowledge in the model search in the form of pre-defined rules.

PyODDS Li et al. (2020b) proposed PyODDS as an end-to-end tool for automatic construction of optimal outlier detection pipelines. The search space consists of 13 anomaly detection algorithms. A hybrid search strategy based on expected improvement is used for optimization. PyODDS assumes an abundance of labels using a fixed split ratio for training and validation set, while the size of the search space is relatively small with most hyperparameters set to fixed values.

MetaOD Zhao et al. (2020) designed MetaOD for the problem of *unsupervised outlier model selection*. MetaOD employs a 2-stage meta-learning approach: (1) an offline phase where a meta-learner is trained on a collection of labeled datasets to extract meta-features and passes them through a matrix factorization to describe the latent meta-space, and (2) an online phase where, given a new unlabeled dataset, the system computes meta-features and predicts the best model configuration based on a dot-product similarity.

4. Applying CASH for Anomaly Detection

When presented with a novel dataset and the task to detect anomalies in that dataset, many anomaly detection algorithms are available to solve this task. Previous research has shown that using a different algorithm or a different configuration of its hyperparameters can result in a significantly improved detection performance (Soenen et al., 2021). It is clear that CASH can guide the search for the best-performing algorithm more efficiently than simple grid search (Zöller and Huber, 2021). Anomaly detection, however, is a fundamentally different problem than the supervised classification and regression tasks that originally served to develop the existing CASH search strategies (Zöller and Huber, 2021). We will now briefly discuss these differences and their implications.

Lack of Label Information Because anomalies in a dataset are typically infrequent and difficult to detect, it is cumbersome to obtain labeled examples of them (Vercruyssen et al., 2018). Therefore, most anomaly detection methods are unsupervised, i.e., they learn a model using a training set without access to the label information (Campos et al., 2016). This is an issue from the perspective of CASH because the model search has to be guided by a clear signal, which is typically the performance of each model configuration on a separate, labeled validation set. In short, the nature of anomaly detection problems imposes a constraint on the maximum size of such a validation set.

In general, increasing the size of the validation set ensures that it is more representative of the true underlying data distribution. Suppose we have two models, A and B, and two validation sets, L(arge) and S(mall) such that $|L| \gg |S|$. If model A outperforms model B on both L and S, the likelihood that model A also outperforms model B on a separate test set is larger for scenario L than for scenario S. Alternatively, if the validation set is too small, the good performance of a model is more likely to be be due to randomness (Soenen et al., 2021). Consider the extreme case of a validation set consisting of only two examples, one *normal* and one *anomaly*. A model that randomly predicts a label for each example still has a 25% probability to obtain perfect accuracy (assuming that the predictions are independent). Adding two more examples to the validation set decreases this probability[2] to 6.25%, and so on. Therefore, a reduced validation set size has an important implication for the CASH search strategies; the smaller the validation set, the noisier the signal that guides the model search. This could hurt the performance of the search strategies.

Label Collection Bias It is well-understood in machine learning that the training and validation set should be i.i.d. samples of the underlying data distribution. This ensures that a learned model generalizes to the unseen test set which, in turn, is also assumed to be an i.i.d. sample of the same data distribution. Because anomalies are infrequent, a random sample of the full dataset is likely to contain only *normal* examples. Suppose that our dataset contains only 2% anomalies and that we randomly label 50 examples to serve as a validation set. Then, the probability of ending up with a validation set containing only normal labels is about 37%. Alternatively, we would need to label at least 35 examples to have a probability ≥50% that one of them is anomalous. A validation set that contains only examples of one class cannot be used to guide the model search, as we cannot compute a loss function. The immediate implication is that our validation set should be collected in a biased manner, such that we can (1) satisfy the size constraints (the validation set cannot be too large), and (2) ensure that it contains at least *some* labeled anomalies such that we can use CASH. This suggest that there is a trade-off between validation set size and label bias in anomaly detection problems.

Search Space Size The search space consists of the algorithms and their configurable hyperparameters. A CASH search strategy learns a function that maps each configuration (A^k, λ^k) of the search space onto its corresponding validation set performance. If the search space becomes larger, either through adding more algorithms or increasing the size of their

2. If the validation set contains n examples, the probability is equal to $(\frac{1}{2})^n$.

hyperparameter domains, the function becomes more difficult to learn (Hutter et al., 2019). Possible solutions are to (1) increase the resource budget $\mathcal{T}$, (2) increase the validation set size, or (3) improve the search strategy. Most anomaly detectors, as opposed to supervised classifiers, have a limited number of hyperparameters. For instance, Lof and Knno have one meaningful hyperparameter; the number of nearest neighbors (Breunig et al., 2000; Ramaswamy et al., 2000).[3] iForest has three meaningful hyperparameters; the number of trees, the sample size per tree, and the number of features used per tree (Liu et al., 2008). Other algorithms, such as Copod (Li et al., 2020c), do not even have hyperparameters to tune. Compare this to the simplest neural network where one can choose the number of layers, the amount of neurons per layer, the activation function per layer, the learning rate, the batch size, and so on. It is possible that larger search spaces increase the usefulness of "smart" guided search strategies over the "naive" unguided ones.

5. Implementing CASH for Anomaly Detection

To enable the use of CASH for selecting the best anomaly detection model for a dataset, we combine the capabilities of Auto-Sklearn and PyOD, two state-of-the-art, open-source Python frameworks. Auto-Sklearn is an AutoML framework that provides functionality for the automatic construction of ML pipelines for supervised classification and regression problems (Feurer et al., 2020). It is built on top of Scikit-Learn[4] and provides, among others, an implementation of the state-of-the-art guided search strategy Smac. PyOD implements dozens of anomaly detection algorithms (Zhao et al., 2019).

Auto-Sklearn does not support unsupervised learning or anomaly detection algorithms by default. For our purposes, we significantly extended[5] its functionality in four ways. First, we included connectors that interface with the anomaly detection algorithms in PyOD. This allows Auto-Sklearn to treat these detectors as any other classifier. Second, we added the option to disable the use of training labels and training loss calculations, which would otherwise prohibit learning unsupervised detectors. Third, we implemented the possibility to create custom labeled validation sets by splitting the original training set according to two search parameters: class prior and size. Finally, we added two additional search strategies to Auto-Sklearn, namely random search and uniform exploration. These can replace the native SMAC search strategy as required. Throughout the experiments we do not use the meta-learning, preprocessing, and ensembling functionality of Auto-Sklearn as these are irrelevant to our experiments.

6. Experiments

The experimental evaluation is structured around the following four questions:

1. Will the guided search strategy Smac outperform the unguided search strategies?

2. Does changing the label bias of the validation set affect how the CASH model selection generalizes to the unseen test data?

3. Based on the PyOD implementation of Lof (Zhao et al., 2019). Other hyperparameters, such as the contamination factor, do not affect Lof's or Knno's fundamental computation of the anomaly score.
4. https://scikit-learn.org/stable/
5. All code is available at: `https://github.com/johnantonn/cash-for-unsupervised-ad`

3. Does increasing the size of the validation set lead to better CASH model selection?
4. Does the performance gap between guided and unguided search strategies increase with an increasing CASH search space?

6.1. Experimental Setup

Search strategies We compare four different search strategies: (1) *random search* (unguided), (2) *uniform exploration* (unguided), (3) Smac (guided), and (4) *default*, a simple baseline that selects the best-performing anomaly detector based on the default hyperparameter settings as set in PyOD. We do not compare to grid search as even simple random search is far superior to this naive strategy (Zöller and Huber, 2021).

Anomaly detectors The CASH search space is comprised of five anomaly detectors: Cblof (He et al., 2003), Copod (Li et al., 2020c), iForest (Liu et al., 2008), Knno (Ramaswamy et al., 2000), and Lof (Breunig et al., 2000). Along with their relevant hyperparameters, they form a conditional space of a total of 19600 configurations, see Table 1. The algorithms were selected because (1) they are radically different in how they detect anomalies, and (2) their individual search space sizes vary substantially.

Setup The experiments are conducted on six datasets listed in Table 2. To reduce run times, we downsampled each dataset to contain at most 5000 examples. Each experiment goes as follows: (1) take a dataset and randomly split it into 75% train set and 25% test set, (2) normalize the features, (3) construct a validation set from the train set, (4) use a search strategy to find the best anomaly detector as measured by the validation set AUC, and (5) apply this detector to the test data and report its test AUC. Each experiment is repeated 10 times and the results are averaged. We use AUC as this is standard in anomaly detection (Campos et al., 2016). In different experiments, we vary the size of the validation set $\in \{20, 50, 100, 200\}$. Additionally, the validation set label bias can be either *stratified* or *balanced*. A stratified validation set mimics the class prior of the full dataset, while a balanced validation set contains an equal amount of normal and anomaly labels (effectively simulating label bias). The total number of experiments is 2100.

Search budget In each experiment, the search budget $\mathcal{T}$ is set to 600 seconds. A sensible value of $\mathcal{T}$ was derived by running a sample of 100 anomaly detector configurations on the datasets, measuring the compute-time, finding the distribution of best-fit and calculating its 90^{th} percentile, so that a single search evaluates at least 1% of the CASH search space.

6.2. Results

6.2.1. Meta-Analysis using Linear Regression

We start with conducting a meta-analysis to estimate the effect of the experimental factors (dataset, validation set size, validation set label bias, and search strategy) on the final test set AUC of the model (i.e., anomaly detector and hyperparameter configuration) selected by each search strategy. We can summarize these results using linear regression models. We identify the significance of each factor by removing one factor at a time and comparing the coefficients of determination (R^2) of the resulting fitted linear regressors to the R^2 of

Table 1: Search space 1: anomaly detectors and hyperparameter domains.

Anomaly Detector	**Hyperparameter Domain**	**Domain Size**

Cblof	alpha={0.5, 0.55, ..., 1.0}
	beta={2, 4, ..., 20}
	contamination={0.05, 0.1, ..., 0.5}
	n_clusters={2, 4, ..., 16}
	use_weights=False
		8800

Copod	contamination={0.05, 0.1, ..., 0.5}	10

iForest	bootstrap=False
	contamination={0.05, 0.1, ..., 0.5}
	max_features=1.0
	max_samples={0.2, 0.25, ..., 1.0}
	n_estimators={5, 10, ..., 200}
		6800

Knno	contamination={0.05, 0.1, ..., 0.5}
	method=largest
	n_neighbors={1, 2, ..., 200}
	p=2
		2000

Lof	contamination={0.05, 0.1, ..., 0.5}
	n_neighbors={1, 2, ..., 200}
	p=2
		2000

Table 2: Summary of the datasets used in the experiments (Campos et al., 2016).

Name	**Datapoints**	**Outliers**	**Outliers %**	**Attributes**
ALOI	50000	1508	3%	27
Annthyroid	7200	534	7%	21
Arrhythmia	450	206	46%	259
Cardiotocography	2126	471	22%	21
SpamBase	4601	1813	39%	57
Waveform	3443	100	3%	21

Table 3: R^2 values for different linear regression models.

Features	$\mathbf{R^2}$	$\mathbf{R^2}$ **loss**
All	0.804	-
w/o dataset	0.186	0.618
w/o search strategy	0.687	0.117
w/o validation set class prior	0.780	0.024
w/o validation set size	0.742	0.062

a regressor fitted using all factors. The target variable (test AUC) was transformed using the *logit* function to map its original $[0, 1]$ range to the reals. The results are presented in Table 3. It is clear that the choice of dataset has the highest impact on predicting the performance, followed by the search strategy, validation set size, and label bias. From the coefficients, we can conclude that a larger validation set contributes positively to test set AUC. Surprisingly, a *balanced* validation set improves test set AUC over its *stratified* variant. A final observation we can make for now, is that *uniform exploration* seems to outperform the other search strategies.

6.2.2. Comparison of Anomaly Detectors

We continue with a comparison of the five anomaly detectors by reporting the percentage of wins of each one for different datasets as well as overall, in Table 4. A win is determined by the anomaly detector that achieves the highest *validation set* AUC score for a specified search strategy, validation set prior and size. The results indicate that there is no single winning algorithm and that the winner is dataset-dependent. This observation coincides with the findings of a recent benchmarking study by Han et al. (2022), which further highlights the importance of model selection for unsupervised anomaly detection tasks.

6.2.3. Q1: Impact of the Search Strategy

Figure 1 presents the average test set AUC of the best incumbent model (i.e., anomaly detector and hyperparameter configuration) over time for each dataset and for different search strategies, using balanced validation sets of size 100. The results show that the *default* search strategy is outperformed by at least one CASH search strategy in each dataset. Further, *uniform exploration* provides the best configuration for 4/6 datasets, while random search and default finish last by finding the worst configuration for 3/6 datasets each.

We apply non-parametric Friedman tests[6] to test whether the different search strategies lead to significantly different results. We run the test for different combinations of validation set label bias and size. The results are presented in Table 5. Especially for small and stratified validation sets, choosing a different search strategy has a significant impact on the final test set AUC. Table 5 also shows the average rank of each search strategy, computed as in (Demšar, 2006). Uniform exploration always achieves the lowest rank, outperforming the other search strategies.

6.2.4. Q2: Impact of the Validation Set Label Bias

To measure the impact of the validation set label bias, we (1) pick a search strategy, dataset, and validation set size, (2) alternate the bias between *balanced* and *stratified*, and (3) compare the resulting test set AUCs. This totals 92 comparisons. The balanced prior leads to a model with higher test set AUC in 73 cases, or 79% of the time. We can conclude that balanced validation sets lead to improved model selection for anomaly detection. This observation might be explained by the fact that balanced validation sets contain more labeled anomalies - which are harder to classify - thus leading to better calibrated anomaly detectors that are less able to overfit on the validation set.

6. We maintain a floating point precision of 3, the critical value of the test is 7.81.

Table 4: Percentage of wins (higher is better) of each anomaly detector, measured by validation set performance, for different datasets and in total.

Dataset	CBLOF	COPOD	IFOREST	KNNO	LOF
ALOI	17.08	6.66	8.75	11.66	**55.83**
Annthyroid	30.83	13.75	12.08	11.66	**31.66**
Arrhythmia	10.83	8.33	**61.66**	10.41	8.75
Cardiotocography	8.33	7.50	**69.58**	7.91	6.66
SpamBase	9.58	**49.16**	24.41	6.25	9.58
Waveform	**37.14**	7.14	30.00	14.28	11.42
All	16.96	17.69	**34.00**	11.75	19.57

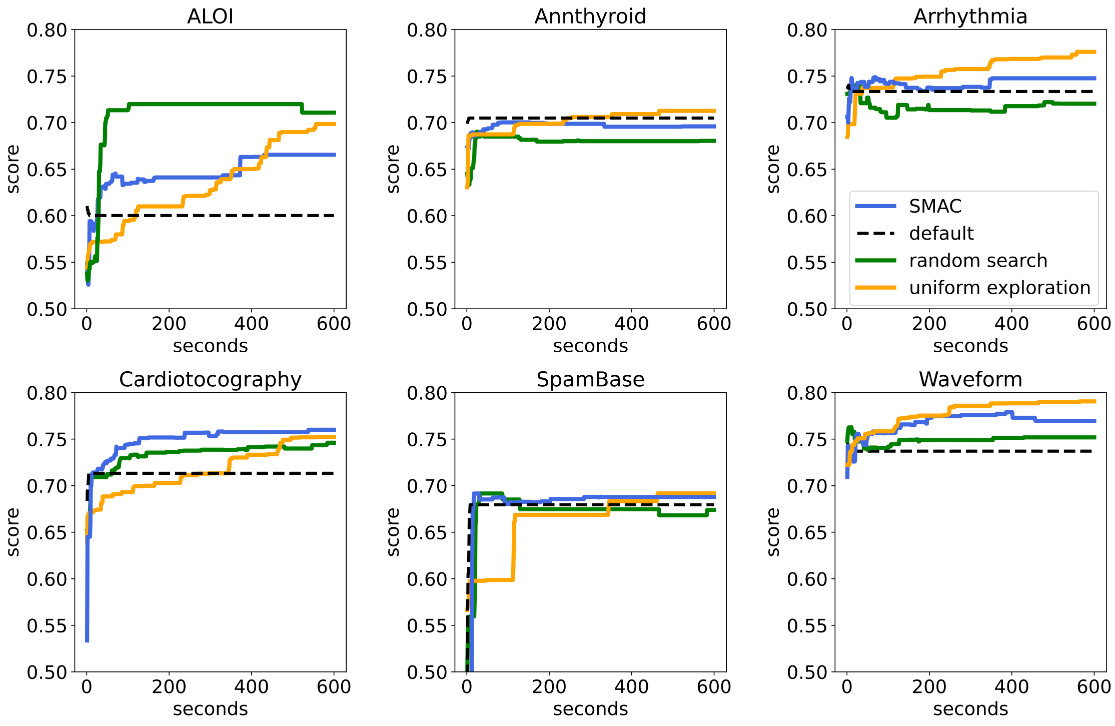

Figure 1: Average test set AUC of the best incumbent model over time for different search strategies, using balanced validation sets of size 100.

Table 5: Average ranks (lower is better) for each search strategy and results of the Friedman tests for different variations of the validation set. P-values indicated with ** are significant. Uniform exploration consistently outperforms the other strategies.

Validation set label bias & size	**Default**	**Random search**	**Uniform exploration**	Smac	**p-value**
Stratified, 20	2.83	2.83	**1.16**	3.16	**0.032****
Stratified, 50	2.83	2.67	**1.00**	3.50	**0.006****
Stratified, 100	2.58	2.75	**1.25**	3.41	**0.031****
Stratified, 200	3.17	2.58	**1.42**	2.83	0.1
Balanced, 20	2.58	3.08	**1.17**	3.17	**0.026****
Balanced, 50	2.92	2.67	**1.42**	3.0	0.12
Balanced, 100	3.33	3.17	**1.33**	2.17	**0.002****
Balanced, 200	3.17	2.17	**1.83**	2.83	0.26

6.2.5. Q3: Impact of the Validation Set Size

To measure the impact of the validation set size, we (1) pick a search strategy, dataset, and validation set label bias, (2) alternate the size $\in \{20, 50, 100, 200\}$, and (3) compare the resulting test set AUCs, totalling 44 comparisons. Validation sets of size 200 win in 32 (73%) of the cases, with sizes of 100 and 50 following with 7 (16%) and 5 (11%) wins, respectively. Figure 2 presents the average validation and test set performance over time achieved by the Smac search strategy using different sizes of balanced validation sets. It is evident from the plots that larger validation sets generalize better. On the contrary, smaller validation sets are prone to overfitting. This result is intuitive when considering that larger validation sets have higher coverage and learn to generalize better on unseen data, while smaller ones might be easier to classify yielding a high performance score, resulting in the promotion of a sub-optimal incumbent.

6.2.6. Q4: Impact of the Search Space Size

To evaluate the impact of the search space size on the test set AUC, a second search space was constructed using the same five anomaly detectors while increasing their hyperparameter domains, resulting in a total size of circa 175 million configurations. Table 6 presents a performance comparison between the two search spaces using the same search budget and balanced validation sets of size 200[7] since these parameters were found to yield the best average test set performance. The results show that an increase in the search space improves the average test set performance for 5 out of 6 datasets.

7. Conclusions

Anomaly detection is generally treated as an unsupervised learning problem since labels are hard to come by. This makes it difficult to select a good performing algorithm for any new

7. For the Waveform dataset, the size was set to 100 since there are not enough anomaly labels to form a balanced validation set of size 200.

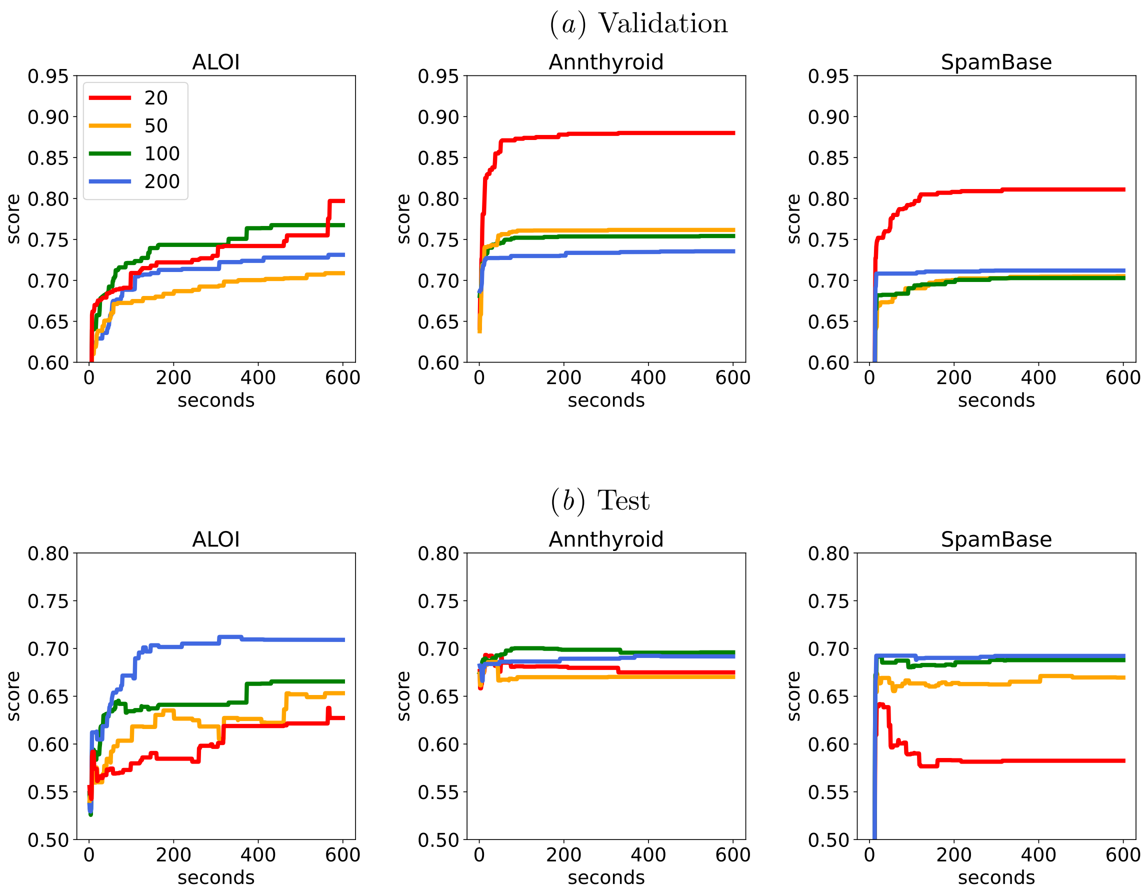

Figure 2: Average validation set AUC (a) and corresponding test set AUC (b) of the best incumbent model found by Smac over time for three datasets (ALOI, Annthyroid, and SpamBase). Reducing the size of the validation set leads to overfitting, which hurts the generalization performance of the selected model.

dataset, especially because dozens of algorithms could be used, each with their own set of hyperparameters. On the other hand, if we are able to collect a small, labeled validation set, we could use it to guide the model search. It is possible to leverage the advances of AutoML research, and in particular the CASH search strategies, for this task. In this paper, we argue that one has to be careful, however, when applying CASH for anomaly detection. Because of the nature of the anomaly detection problem, the collected validation set will be small and biased. We make three observations. First, the smaller the validation set, the less useful it is to guide the model search. Second, biased validation sets (i.e., including an equal amount of anomalous and normal examples) guide the model search toward models that generalize better to unseen test sets. Finally, increasing the search space size might improve the model search.

Table 6: Best average test set AUC achieved per dataset for each search space. The table also includes the average test set AUC for the *default* search strategy as a comparison.

Dataset	**Default**	**SP1**	**SP2**	**(SP2-SP1)/SP1 %**
ALOI	0.605	0.714	0.727	1.8
Annthyroid	0.706	0.724	0.874	20.7
Arrhythmia	0.746	0.776	0.775	0.0
Cardiotocography	0.720	0.767	0.819	6.8
SpamBase	0.692	0.695	0.739	6.3
Waveform	0.737	0.797	0.812	1.9

Acknowledgements

This work is supported by the Flemish government “Onderzoeksprogramma Artificiële Intelligentie Vlaanderen” (JD), by iBOF/21/075 (JD), and the “Agentschap Innoveren & Ondernemen (VLAIO)” (VV) as part of the innovation mandate HBC.2020.2297.

References

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305, feb 2012. ISSN 1532-4435.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

Guilherme O. Campos, Arthur Zimek, Jörg Sander, Ricardo J. G. B. Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E. Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, Jul 2016. ISSN 1573-756X. doi: 10.1007/s10618-015-0444-8. URL https://doi.org/10.1007/s10618-015-0444-8.

Sayan Chakraborty, Smit Shah, Kiumars Soltani, Anna Swigart, Luyao Yang, and Kyle Buckingham. Building an automated and self-aware anomaly detection system. *CoRR*, abs/2011.05047, 2020. URL https://arxiv.org/abs/2011.05047.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, dec 2006. ISSN 1532-4435.

Rémi Domingues, Maurizio Filippone, Pietro Michiardi, and Jihane Zouaoui. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern recognition*, 74:406–421, 2018.

Andrew Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern, and Weng-Keen Wong. A meta-analysis of the anomaly detection problem. *arXiv preprint arXiv:1503.01158*, 2015.

Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Auto-sklearn 2.0: The next generation. *arXiv preprint arXiv:2007.04074*, 24, 2020.

Songqiao Han, Xiyang Hu, Hailiang Huang, Mingqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark, 2022. URL https://arxiv.org/abs/2206.09426.

Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650, 2003.

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In Carlos A. Coello Coello, editor, *Learning and Intelligent Optimization*, pages 507–523, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-25566-3.

Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. An efficient approach for assessing hyperparameter importance. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 754–762, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/hutter14.html.

Frank Hutter, Joaquin Vanschoren, and Lars Kotthoff. *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019.

Kwei-Herng Lai, Daochen Zha, Guanchu Wang, Junjie Xu, Yue Zhao, Devesh Kumar, Yile Chen, Purav Zumkhawaka, Minyang Wan, Diego Martinez, and Xia Hu. TODS: an automated time series outlier detection system. *CoRR*, abs/2009.09822, 2020. URL https://arxiv.org/abs/2009.09822.

Yuening Li, Zhengzhang Chen, Daochen Zha, Kaixiong Zhou, Haifeng Jin, Haifeng Chen, and Xia Hu. Autood: Automated outlier detection via curiosity-guided search and self-imitation learning. *CoRR*, abs/2006.11321, 2020a. URL https://arxiv.org/abs/2006.11321.

Yuening Li, Daochen Zha, Praveen Kumar Venugopal, Na Zou, and Xia Hu. Pyodds: An end-to-end outlier detection system with automated machine learning. *CoRR*, abs/2003.05602, 2020b. URL https://arxiv.org/abs/2003.05602.

Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. COPOD: Copula-based outlier detection. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, nov 2020c. doi: 10.1109/icdm50108.2020.00135. URL https://doi.org/10.1109%2Ficdm50108.2020.00135.

Marius Lindauer, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter. Smac3: A versatile bayesian optimization package for hyperparameter optimization. *Journal of Machine Learning Research*, 23(54):1–9, 2022. URL http://jmlr.org/papers/v23/21-0888.html.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5(1):18, May 2016. ISSN 2192-6670. doi: 10.1007/s13721-016-0125-6. URL https://doi.org/10.1007/s13721-016-0125-6.

Tahereh Pourhabibi, Kok-Leong Ong, Booi H Kam, and Yee Ling Boo. Fraud detection: A systematic literature review of graph-based anomaly detection approaches. *Decision Support Systems*, 133:113303, 2020.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438, 2000.

Pieter Robberechts, Maarten Bosteels, Jesse Davis, and Wannes Meert. Query log analysis: Detecting anomalies in dns traffic at a tld resolver. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 55–67. Springer, 2018.

Jonas Soenen, Elia Van Wloputte, Lorenzo Perini, Vincent Vercruyssen, Wannes Meert, Jesse Davis, and Hendrick Blockeel. The effect of hyperparameter tuning on the comparative evaluation of unsupervised anomaly detection methods, 2021. URL https://people.cs.kuleuven.be/~lorenzo.perini/files/HyperparametersAD.pdf.

Vincent Vercruyssen, Wannes Meert, Gust Verbruggen, Koen Maes, Ruben Baumer, and Jesse Davis. Semi-supervised anomaly detection with an application to water analytics. In *2018 ieee international conference on data mining (icdm)*, volume 2018, pages 527–536. IEEE, 2018.

Yue Zhao, Zain Nasrullah, and Zheng Li. Pyod: A python toolbox for scalable outlier detection. *CoRR*, abs/1901.01588, 2019. URL http://arxiv.org/abs/1901.01588.

Yue Zhao, Ryan A. Rossi, and Leman Akoglu. Automating outlier detection via meta-learning. *CoRR*, abs/2009.10606, 2020. URL https://arxiv.org/abs/2009.10606.

Marc-André Zöller and Marco F. Huber. Benchmark and survey of automated machine learning frameworks. *J. Artif. Int. Res.*, 70:409–472, may 2021. ISSN 1076-9757. doi: 10.1613/jair.1.11854. URL https://doi.org/10.1613/jair.1.11854.