N-1 Experts: Unsupervised Anomaly Detection Model Selection

Constantin Le Cleï[1,2] **Yasha Pushak**[1] **Fatjon Zogaj**[1,2] **Moein Owhadi-Kareshk**[1] **Zahra Zohrevand**[1] **Robert Harlow**[1,3] **Hesam Fathi Moghadam**[1] **Sungpack Hong**[1] **Hassan Chafi**[1]

[1]Oracle Labs
[2]ETH Zürich
[3]UW Madison

Abstract Manually finding the best combination of machine learning training algorithm, model and hyperparameters can be challenging. In supervised settings, this burden has been alleviated with the introduction of automated machine learning (AutoML) methods. However, similar methods are noticeably absent for fully unsupervised applications, such as anomaly detection. We introduce one of the first such methods, N-1 Experts, which we compare to a recent state-of-the-art baseline, MetaOD, and show favourable performance.

1 Introduction

In the last several decades, supervised machine learning has attracted substantial attention due to its state-of-the-art performance on a wide variety of applications (for example, see Golden (2017); Arcadu et al. (2019); Voyant et al. (2017); Angermueller et al. (2016) or Chen et al. (2018)). To solve these problems, numerous different machine learning methods have been proposed (for example, see Cortes and Vapnik (1995); Breiman (2001) or Chen and Guestrin (2016)). However, the widely-known *no free lunch theorem* (Wolpert and Macready, 1995, 1997) implies that there is no single best method for all datasets. To address this challenge, methods for automatically selecting the best algorithm for a given problem instance have been proposed for numerous applications (Xu et al., 2008; Kotthoff et al., 2015; Kerschke et al., 2019; Abell et al., 2012; Belkhir et al., 2016; Malan, 2018). This problem is further complicated in machine learning (ML), where model performance is known to depend strongly on the performance of its training algorithm's hyperparameters (Bergstra et al., 2011; Bergstra and Bengio, 2012), which has lead to a plethora of methods for *model selection*, and more generally, *combined algorithm selection and hyperparameter optimzization (CASH)* (Thornton et al., 2013) (for example, see Li et al. (2020a); Hutter et al. (2011); Semenkina and Semenkin (2014); Parker-Holder et al. (2020); Yuan et al. (2021); Li et al. (2021); Pushak and Hoos (2020); Yakovlev et al. (2020) or Lindauer et al. (2022)). These CASH techniques have proven very successful, but require labelled data in order to train and evaluate model configurations. While little labelled data is needed for good performance in some cases (Zogaj et al., 2021), requiring any labels at all makes these methods unsuitable for CASH on unsupervised ML problems, such as anomaly detection, where the labels are unknown.

In *unsupervised anomaly detection (UAD)*, the goal is to identify which data instances do not belong in the same distribution as the majority of the data in a given dataset. The use-cases for anomaly detection are pervasive. For example, two particularly prominent applications include medicine (Schlegl et al., 2017; Wong et al., 2003; Hauskrecht et al., 2007) and security (Hu et al., 2003; Vanerio and Casas, 2017). In both cases, correctly identifying relevant anomalies is of the utmost importance. In medical applications, failing to detect important anomalies could result in potentially

[2, 3] Research was conducted during internships at Oracle Labs.

life-threatening diseases going untreated. In security applications, incorrectly flagging benign activity as anomalous could be equally harmful, as it may result in the unfair denial-of-service – or even prosecution – of the innocent.

One of the earliest methods in the field proposed to use Cook's distance to detect anomalies (Cook, 1977). Since then, similar to supervised machine learning, a diverse set of anomaly detection methods have been proposed to solve this problem. These include: methods that assume data-linearity (Hardin and Rocke, 2004; Rousseeuw and Driessen, 1999; Shyu et al., 2003), proximity-based methods (Ramaswamy et al., 2000; Goldstein and Dengel, 2012; He et al., 2003; Almardeny et al., 2020), probabilistic methods (Crosby, 1994; Li et al., 2020b), ensemble-based methods (Liu et al., 2008; Lazarevic and Kumar, 2005; Zhao et al., 2019a), boundary-based methods (Schölkopf et al., 2001; Tax and Duin, 2004; Ruff et al., 2018; Zhou et al., 2021) and neural-network-based methods (An and Cho, 2015; Zhou and Paffenroth, 2017; Goodfellow et al., 2014)

To the best of our knowledge – and despite the wide variety of anomaly detection methods – there is only a single existing model selection method for unsupervised anomaly detection: MetaOD (Zhao et al., 2021). By extension, this is also the first method for the unsupervised CASH problem, as we can consider different configurations of a single model as if they were different models. MetaOD extracts high-level meta-features from a collection of historical anomaly detection datasets (with known labels). It then trains and evaluates a large number of anomaly detection models with varying hyperparameter configurations on each such dataset. Using this collection of model performance data, MetaOD trains a supervised machine learning model to predict which model and hyperparameter configuration should perform best, given the meta-features of a new anomaly detection dataset.

In this work, we introduce N-1 Experts, which is – to the best of our knowledge – the first *fully* unsupervised UAD model selection/CASH method (see Section 2). That is, in contrast to MetaOD, N-1 Experts does not require labelled datasets for meta-learning. Instead, similar to ensembling algorithms (Zimek et al., 2014; Aggarwal, 2013; Vanerio and Casas, 2017) N-1 Experts leverages complementary strengths between candidate models to select the model, that is most similar to the others. We find that N-1 Experts usually outperforms the current state-of-the-art method across multiple settings. We hypothesized that N-1 Experts should perform best when selecting between a relatively small set of high-quality candidate models, but to our surprise, N-1 Experts seems to work equally well for selecting between random configurations (see Section 3). This raises additional interesting questions and promising directions for future work (see Section 5).

2 Methods

Problem Definition (UAD model selection). Let X be a matrix that contains features as the columns and data instances or samples as rows. Let $y_i \in \{0, 1\}$ be the corresponding unknown label for each row x_i of X, such that $y_i = 1$ indicates that row x_i is an anomaly. We define the contamination factor, c, to be the percentage of rows in X that are labelled as anomalies in y. Let A be a set of unsupervised anomaly detection algorithms. That is, each $a \in A$ is a training algorithm that accepts as input a dataset X and then produces as output a trained model m. Each trained model m is a function that can map from a data instance, x_i, to an outlier score $\hat{o}_i \in \mathbb{R}$. Larger values of $\hat{o}_i$ correspond to a stronger prediction that x_i is anomalous. Given an estimate for the contamination factor, c, the set of outlier scores for X can be mapped into a set of predicted labels $\hat{y}$.

Our goal is to identify the training algorithm $a^* \in A$ that produces the model m^* that is the most accurate (in term of some metric d) at predicting the true anomaly labels y, when given the true contamination factor, c. We refer to this problem as *UAD model selection* and propose the N-1 Experts framework to solve it. For simplicity, throughout the following paper we will omit mentioning training algorithms A and simply refer to a set of candidate models M, which have been trained with their corresponding algorithms and hyperparameter configurations on dataset X.

In most anomaly detection applications, the contamination factor is assumed to be known. However, in practice, it is often difficult to know in advance how many anomalies are present in a given dataset. For this reason, our framework is agnostic to the true contamination factor.

Proposed Framework. N-1 Experts overcomes the challenge of having no labels by taking advice from a group of "expert" models. That is, our framework evaluates candidate models based on how closely their predictions align to each other candidate model. As shown in Algorithm 1 in Appendix A, in N-1 Experts, each candidate model will alternatively be evaluated and used to evaluate other models. The main procedure takes as input a dataset X with $n \in \mathbb{N}$ samples, a set M of models trained on X, a metric d and a custom set, C of contamination factors, which should be in $(0, 0.5]$. N-1 Experts consists of two phases: pseudo labelling and model evaluation.

1. **Pseudo Labelling**. For each candidate model $m \in M$ and each contamination factor $c \in C$, a set of pseudo labels $L_m(c)$ are assigned to the training set as follows: the $n \cdot c$ training points with the highest outlier scores $\hat{o}$ (as predicted by model m) are labelled anomalies, and the remaining points are considered normal. Since the true contamination factor is unknown in real environments, the N-1 Experts framework provides an estimate of the metric d by averaging over the set of contamination factors C. However, if the true contamination factor, c^* is known, we only have one set of pseudo labels $L_m(c^*)$ for each model.

2. **Model Evaluation**. For a given expert model m' and each contamination factor c in C, we compute using metric d the score of model m on the dataset X with respect to the artificial labels $L_{m'}(c)$. We then define the *expert score* for model m with respect to expert m' as the average of these scores across contamination factors. We repeat this process for all expert models $m' \in M \backslash \{m\}$, and aggregate these expert scores (using any aggregation method, like the mean) to produce the *aggregated score*, $\mathbb{S}[m]$, for model m. Finally, the candidate $\hat{m}$ with the largest aggregated score is selected.

3 Experimental Setup

To the best of our knowledge, MetaOD (Zhao et al., 2021) is the only existing method designed for UAD model selection, hence we use it as a baseline to measure N-1 Experts's performance. Zhao et al. (2021) released a pre-trained version of MetaOD online. However, we do not use it because we need to select between different sets of candidate models (see below). Instead, we retrain MetaOD (using the open-source implementation and keeping the default hyperparameters) from scratch for each dataset. Similar to Zhao et al. (2021), we use a leave-one-out procedure, that employs all but one dataset for meta-learning and then predicts the best model for the last one. Among the datasets used for meta-learning, MetaOD uses some of the datasets for meta-training and the others for selecting some of its hyperparameters. We create these splits randomly using 85% of the datasets for meta-training and 15% of them for validation. For N-1 Experts, we set the metric, d, to be AUROC and the set, C, of contaminators to be a set of 10 evenly spaced points between 0.01 and 0.5. In addition to the two selection methods, MetaOD and N-1 Experts, we also report the performance of Single Best, which corresponds to always selecting the single model with the best average performance on a given set of datasets. We evaluate each method on two sets of datasets (see **ODDS**, **SSL** below). When evaluating Single Best on a given set of datasets we differentiate between picking the model that performs best on that same set of datasets, Single Best$_{(\text{SD})}$, and picking the model that performs best on the other set of datasets, Single Best$_{(\text{OD})}$.

In order to cover various UAD model selection scenarios, we diversify our choice of datasets and candidate models. A high-level summary of each dataset is given in Appendix B. Our datasets can be separated into two sets as follows:

- **ODDS**: 14 diverse datasets from the Stonybrook outlier detection datasets (Rayana, 2016); and,

- **SSL**: 8 datasets consisting of software security logs by one of our industry partners[1].

Furthermore, we consider two different sets of candidate models:

- Optimized Configurations (**OC**): A set of 8 candidate models from the PyOD library (Zhao et al., 2019b), for which the hyperparameters have been configured to obtain strong average performance across a range of different anomaly detection datasets. We selected these configurations using a similar procedure as the one described in Section 3.1 of (Yakovlev et al., 2020). These models therefore correspond to a relatively reasonable initialization and can be considered acceptable proxies for "experts".
- Random Configurations (**RC**): We also consider a set of models sampled from the 302 model configurations used by Zhao et al. (2021) to benchmark MetaOD. This original set is highly unbalanced; for example, there are over eleven times as many IsolationForest (Liu et al., 2008) hyperparameter configurations than there are for ABOD (Kriegel et al., 2008). As a result, N-1 Experts would almost always select one of the IsolationForest models rather than any of the others. To avoid this bias, we only included five different random configurations of each model type in this set.

A complete description of each of those model sets is given in Appendix C. Since the average model in **RC** is likely of lower quality than in **OC**, we hypothesize that N-1 Experts will perform worse using **RC** as opposed to **OC** for its candidate models.

Given an experimental scenario (a tuple that defines the set of datasets and models), each selection method chooses a candidate model for each dataset. To compare performance on a dataset we use AUROC regret, where the regret is calculated as the difference in AUROC between the highest scoring model and the selected one. Finally, we perform 15 independent runs of each of the 4 experimental scenarios, which constitute the cross product of the sets of datasets and models.

4 Results

In Table 1, we show the mean AUROC regrets for the methods defined in Section 3. Note that, when evaluating performance for example on the **ODDS** datasets, Single Best$_{(SD)}$ refers to choosing the model that performed best on the *same* **ODDS** datasets, while Single Best$_{(OD)}$ refers to the model that performed best on the *other* set of datasets, **SSL**.

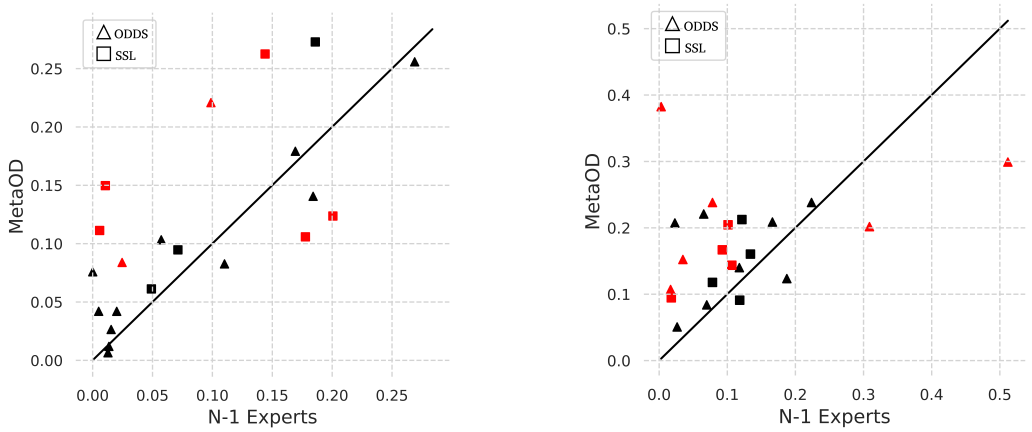
In all four experimental scenarios, N-1 Experts outperforms MetaOD according to a Wilcoxon signed rank test with a 5% significance level. Surprisingly, both N-1 Experts and MetaOD perform worse than picking the single best model for each experimental scenario when the model is chosen to be the best among the same set of datasets (**SD**). However, the single best model is not consistent between each experimental setup. Indeed, while N-1 Experts performs worse than Single Best$_{(SD)}$, it performs better than Single Best$_{(OD)}$ for three out of the four experiments. The comparison against Single Best$_{(OD)}$ is fairer, because Single Best$_{(SD)}$ has the advantage that the models' real scores on the test datasets are used to help pick the selected model. Nevertheless, in practice, picking a single best model may work well if all datasets are relatively similar; however, this result clearly demonstrates why a more intelligent selection method should be preferred.

To provide a more detailed comparison, we show a scatter plot comparing the regrets of N-1 Experts and MetaOD on each dataset in Figure 1. Note that both selection methods pick out of the same set of models, such that the regrets are computed with respect to the same highest scoring model. We can see that N-1 Experts frequently performs better than MetaOD in all four experimental scenarios. For model set **OC** (containing models with optimized high-quality default configurations), N-1 Experts achieves lower regret on 14 out of 22 dataset (or 5 out of 7 datasets for the results that are statistically significant) and for model set **RC** (containing models with

[1]Unfortunately, due to the nature of these datasets, we are unable to make them publically available.

Table 1: The mean regret on all datasets of a given set, across 15 different independent runs of each method. The 0.025 and 0.975 quantiles of the regret are shown in brackets. The results from N-1 Experts, MetaOD and Single Best$_{(OD)}$ are shown in boldface, if they are not worse than the best of the three methods for a given experimental scenario, according to a Wilcoxon signed rank test. Note that we exclude Single Best$_{(SD)}$ from the comparison with statistical tests, because it has the unfair advantage of knowing which model actually performs best on the test datasets.

Method	Optimized Configurations (**OC**)		Random Configurations (**RC**)	
	ODDS	**SSL**	**ODDS**	**SSL**
N-1 Experts	**0.090** [0.084, 0.102]	**0.106** [0.082, 0.142]	0.131 [0.106, 0.166]	**0.096** [0.051, 0.181]
MetaOD	0.104 [0.062, 0.138]	0.148 [0.086, 0.241]	0.190 [0.133, 0.322]	0.149 [0.072, 0.214]
Single Best$_{(OD)}$	0.129 [0.127, 0.131]	0.219 [0.213, 0.231]	**0.117** [0.116, 0.118]	0.130 [0.090, 0.345]
Single Best$_{(SD)}$	0.060 [0.056, 0.069]	0.060 [0.052, 0.070]	0.095 [0.090, 0.102]	0.055 [0.033, 0.071]



(a) Mean AUROC regret Expert Models (**OC**)

(b) Mean AUROC regret Random Models (**RC**)

Figure 1: Individual dataset regrets plotted as the mean over 15 independent runs. Points above the line correspond to better performance with N-1 Experts. Datasets are marked in red if the performance difference is significant at a 5% significance level according to a Wilcoxon signed rank text.

random configurations) this fraction improves to 18 out of 22 datasets (or 8 out of 10 datasets for the significant results).

We had hypothesized that N-1 Experts could perform worse on the **RC** set than on the **OC** set. However, the results in Table 1 do not support this; in fact, these results indicate that N-1 Experts outperforms MetaOD in both scenarios. Similarly, we had speculated that MetaOD would perform best on the software security logs (**SSL**) datasets, because each dataset comes from similar sources. However, we again see that this does not appear to be true, signifying N-1 Experts' strong versatility with respect to different models and datasets.

Furthermore, we observe that MetaOD yields higher variance between runs (yielding larger performance variance, see Table 1 and Appendix D), which could partly explain these observations. MetaOD's larger variance may be due to the relatively small number of datasets available to it during its meta-learning stage, which leads to a high variance when randomly picking its training and validation sets (see Section 3). On the other hand, since the only source of variance in N-1

Experts is the stochasticity of the anomaly detection training algorithms, the latter is much more consistent.

In contrast with meta-learning-based approaches, N-1 Experts does not require any pre-training on historical datasets, and therefore it does not incur an offline running time cost. However it does require more running time to select a model, since all of the candidate models must be trained on the dataset at selection time. We show a comparison of these running times in Appendix E.

5 Conclusions and Future Work

We have introduced a novel model selection method, N-1 Experts, for unsupervised anomaly detection that exploits complementary strengths between a set of candidate models. As opposed to meta-learning-based approaches, N-1 Experts does not require labelled historical datasets for pre-training. It is therefore fully unsupervised, and can be applied even when users may not know the correct labels. Our experiments on diverse sets of datasets and models show that N-1 Experts generally performs better and more consistently than the current state of the art.

While promising, not all of our results were expected. We had hypothesized that N-1 Experts would perform best when selecting between high-quality models; however, in practice we observed that it suffered no performance degradation when given random configurations. Similarly, we had anticipated that MetaOD would have the strongest advantage on the **SSL** datasets, since they came from similar applications. However, we again found no such evidence. Indeed, both unanticipated outcomes open more questions that could be addressed in future work, in which it would also be beneficial to compare N-1 Experts with ensembling-based approaches (Vanerio and Casas, 2017).

By design, we studied the simplest implementation of N-1 Experts. As a result, N-1 Experts required a balanced set of models. However, additional sophistication may further improve its performance. For example, it may be possible to use a set of historical datasets to learn a set of weights that correspond to how much each expert model should be trusted. Alternatively, using the median or another quantile to aggregate the experts scores (instead of the mean), could help mitigate the effects of poor-quality expert models.

Finally, the strong performance of N-1 Experts even on random model configurations indicates that it could also be used as a proxy metric for iterated hyperparameter configuration methods. In that case, one could either use a predefined pool of experts, or dynamically add and remove new experts over time.

References

Abell, T., Malitsky, Y., and Tierney, K. (2012). *Fitness landscape based features for exploiting black-box optimization problem structure.* IT University of Copenhagen.

Aggarwal, C. C. (2013). Outlier ensembles: position paper. *ACM SIGKDD Explorations Newsletter*, 14(2):49–58.

Almardeny, Y., Boujnah, N., and Cleary, F. (2020). A novel outlier detection method for multivariate data. *IEEE Transactions on Knowledge and Data Engineering*, 32.

An, J. and Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18.

Angermueller, C., Pärnamaa, T., Parts, L., and Stegle, O. (2016). Deep learning for computational biology. *Molecular Systems Biology*, 12(7).

Arcadu, F., Benmansour, F., Maunz, A., Willis, J., Haskova, Z., and Prunotto, M. (2019). Deep learning algorithm predicts diabetic retinopathy progression in individual patients. *NPJ Digital Medicine*, 2(1):1–9.

Belkhir, N., Dréo, J., Savéant, P., and Schoenauer, M. (2016). Feature based algorithm configuration: A case study with differential evolution. In *Proceedings of the Fourteenth International Conference on Parallel Problem Solving from Nature (PPSN 2016)*, pages 156–166.

Bergstra, J. S., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Proceedings of the Twenty-Fifth Conference on Advances in Neural Information Processing Systems (NeurIPS 2011)*, pages 2546–2554.

Bergstra, J. S. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(10):281–305.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breunig, M. M., Kriegel, H., Ng, R. T., and Sander, J. (2000). LOF: identifying density-based local outliers. In Chen, W., Naughton, J. F., and Bernstein, P. A., editors, *Proceedings of the Twenty-Ninth ACM SIGMOD International Conference on Management of Data (SIGMOD 2000)*, pages 93–104. ACM.

Chen, H., Engkvist, O., Wang, Y., Olivecrona, M., and Blaschke, T. (2018). The rise of deep learning in drug discovery. *Drug Discovery Today*, 23(6):1241–1250.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the Twenty-Second ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD 2016)*, pages 785–794.

Chollet, F. et al. (2015). Keras.

Cook, R. D. (1977). Detection of influential observation in linear regression. *Technometrics*, 19(1):15–18.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Crosby, T. (1994). How to detect and handle outliers. *Technometrics*, 36(3).

Golden, J. A. (2017). Deep learning algorithms for detection of lymph node metastases from breast cancer: helping artificial intelligence be seen. *Journal of the American Medical Association (JAMA)*, 318(22):2184–2186.

Goldstein, M. and Dengel, A. (2012). Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. In *Proceedings of the Thirty-Fifth German Conference on Artificial Intelligence (KI 2012)*. Citeseer.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Proceedings of the Twenty-Seventh Conference on Advances in Neural Information Processing Systems (NeurIPS 2014)*.

Hardin, J. and Rocke, D. M. (2004). Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Computational Statistics and Data Analysis*, 44(4):625–638.

Hauskrecht, M., Valko, M., Kveton, B., Visweswaran, S., and Cooper, G. F. (2007). Evidence-based anomaly detection in clinical domains. In *Proceedings of the 2007 American Medical Informatics Association (AMIA) Annual Symposium*, page 319.

He, Z., Xu, X., and Deng, S. (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10):1641–1650.

Hu, W., Liao, Y., and Vemuri, V. R. (2003). Robust support vector machines for anomaly detection in computer security. In *Proceedings of the First IEEE International Conference on Machine Learning and Applications (ICMLA 2003)*, pages 168–174.

Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *Proceedings of the Fifth Learning and Intelligent Optimization Conference (LION 2011)*, Lecture Notes in Computer Science (LNCS), pages 507–523.

Kerschke, P., Hoos, H. H., Neumann, F., and Trautmann, H. (2019). Automated algorithm selection: Survey and perspectives. *Evolutionary Computation*, 27(1):3–45.

Kotthoff, L., Kerschke, P., Hoos, H., and Trautmann, H. (2015). Improving the state of the art in inexact TSP solving using per-instance algorithm selection. In *Proceedings of the Ninth International Conference on Learning and Intelligent Optimization (LION 2015)*, pages 202–217.

Kriegel, H.-P., Schubert, M., and Zimek, A. (2008). Angle-based outlier detection in high-dimensional data. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD 2008)*, pages 444–452.

Lazarevic, A. and Kumar, V. (2005). Feature bagging for outlier detection. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (SIGKDD 2005)*, pages 157–166.

Li, W., Wang, T., and Ng, W. W. (2021). Population-based hyperparameter tuning with multitask collaboration. *IEEE Transactions on Neural Networks and Learning Systems*.

Li, Y., Jiang, J., Gao, J., Shao, Y., Zhang, C., and Cui, B. (2020a). Efficient automatic CASH via rising bandits. In *Proceedings of the Thirty-fourth AAAI Conference on Artificial Intelligence (AAAI 2020)*, number 04, pages 4763–4771.

Li, Z., Zhao, Y., Botta, N., Ionescu, C., and Hu, X. (2020b). Copod: copula-based outlier detection. In *Proceedings of the Twentieth IEEE International Conference on Data Mining (ICDM 2020)*, pages 1118–1123. IEEE.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., and Hutter, F. (2022). Smac3: A versatile bayesian optimization package for hyperparameter optimization. *Journal of Machine Learning Research*, 23(54):1–9.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM 2008)*, pages 413–422. IEEE.

Malan, K. M. (2018). Landscape-aware constraint handling applied to differential evolution. In *Proceedings of the Seventh International Conference on Theory and Practice of Natural Computing (TPNC 2018)*, pages 176–187.

Parker-Holder, J., Nguyen, V., and Roberts, S. J. (2020). Provably efficient online hyperparameter optimization with population-based bandits. In *Proceedings of the Thirty-Third International Conference on Advances in Neural Information Processing Systems (NeurIPS 2020)*, pages 17200–17211.

Pevný, T. (2016). Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102(2):275–304.

Pushak, Y. and Hoos, H. H. (2020). Golden parameter search: Exploiting structure to quickly configure parameters in parallel. In *Proceedings of the Twenty-second International Conference on Genetic and Evolutionary Computation (GECCO 2020)*, pages 245–253.

Ramaswamy, S., Rastogi, R., and Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In *Proceedings of the Twenty-Ninth ACM SIGMOD International Conference on Management of Data (SIGMOD 2000)*, pages 427–438.

Rayana, S. (2016). ODDS library.

Rousseeuw, P. J. and Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., Müller, E., and Kloft, M. (2018). Deep one-class classification. In *Proceedings of the Thirty-Fifth International Conference on Machine Learning (ICML 2018)*, pages 4393–4402. PMLR.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., and Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *Proceedings of the Twenty-Fifth International Conference on Information Processing in Medical Imaging (IPMI 2017)*, pages 146–157. Springer.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471.

Semenkina, M. and Semenkin, E. (2014). Hybrid self-configuring evolutionary algorithm for automated design of fuzzy classifier. In *Proceedings of the Fifth International Conference on Advances in Swarm Intelligence (ICSI 2014)*, pages 310–317. Springer.

Shyu, M.-L., Chen, S.-C., Sarinnapakorn, K., and Chang, L. (2003). A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the Workshop on IEEE Foundations and New Directions of Data Mining, in conjunction with the Third IEEE International Conference on Data Mining (ICDM 2003)*.

Tax, D. M. and Duin, R. P. (2004). Support vector data description. *Machine learning*, 54(1):45–66.

Thornton, C., Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2013). Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the Nineteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD 2013)*, pages 847–855.

Vanerio, J. and Casas, P. (2017). Ensemble-learning approaches for network security and anomaly detection. In *Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks (Big-DAMA 2017), in conjunction with the Forty-seventh Conference of the ACM Special Interest Group on Data Communication (SIGCOMM 2017)*, pages 1–6.

Voyant, C., Notton, G., Kalogirou, S., Nivet, M.-L., Paoli, C., Motte, F., and Fouilloy, A. (2017). Machine learning methods for solar radiation forecasting: A review. *Renewable Energy*, 105:569–582.

Wolpert, D. H. and Macready, W. G. (1995). No free lunch theorems for search. Technical report, SFI-TR-95-02-010, Santa Fe Institute.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

Wong, W.-K., Moore, A. W., Cooper, G. F., and Wagner, M. M. (2003). Bayesian network anomaly pattern detection for disease outbreaks. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003)*, pages 808–815.

Xu, L., Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2008). SATzilla: portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32:565–606.

Yakovlev, A., Moghadam, H. F., Moharrer, A., Cai, J., Chavoshi, N., Varadarajan, V., Agrawal, S. R., Idicula, S., Karnagel, T., Jinturkar, S., and Agarwal, N. (2020). Oracle AutoML: a fast and predictive AutoML pipeline. *Proceedings of the Forty-Sixth International Conference on Very Large Data Bases (VLDB 2020)*, 13(12):3166–3180.

Yuan, Y., Wang, W., and Pang, W. (2021). A genetic algorithm with tree-structured mutation for hyperparameter optimisation of graph neural networks. In *Proceedings of the Twenty-Third IEEE Congress on Evolutionary Computation (CEC 2021)*, pages 482–489.

Zhao, Y., Nasrullah, Z., Hryniewicki, M. K., and Li, Z. (2019a). Lscp: Locally selective combination in parallel outlier ensembles. In *Proceedings of the Twentieth SIAM International Conference on Data Mining (SDM 2019)*, pages 585–593.

Zhao, Y., Nasrullah, Z., and Li, Z. (2019b). PyOD: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, 20(96):1–7.

Zhao, Y., Rossi, R., and Akoglu, L. (2021). Automatic unsupervised outlier model selection. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Proceedings of the Thirty-Fourth Conference on Advances in Neural Information Processing Systems (NeurIPS 2021)*, pages 4489–4502. Curran Associates, Inc.

Zhou, C. and Paffenroth, R. C. (2017). Anomaly detection with robust deep autoencoders. In *Proceedings of the Twenty-Third ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD 2017)*, pages 665–674.

Zhou, Y., Liang, X., Zhang, W., Zhang, L., and Song, X. (2021). Vae-based deep svdd for anomaly detection. *Neurocomputing*, 453:131–140.

Zimek, A., Campello, R. J., and Sander, J. (2014). Ensembles for unsupervised outlier detection: challenges and research questions a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):11–22.

Zogaj, F., Cambronero, J. P., Rinard, M. C., and Cito, J. (2021). Doing more with less: Characterizing dataset downsampling for automl. In *Proceedings of the Forty-Seventh International Conference on Very Large Data Bases (VLDB 2021)*, pages 2059–2072.

A N-1 Experts

In Figure 2, we show an example of the calculation for our proposed framework N-1 Experts and in Algorithm 1, we show the pseudo code of N-1 Experts.

Algorithm 1 N-1 Experts

Input: Trained models, M; contamination factors, C; dataset, X; metric, d.
Output: Estimate $\hat{m} \in M$ of the best model with respect to metric d.

procedure N-1 Experts(M, C, X, d)
  for m in M ▷ Pseudo Labelling
    for c in C
      $L_m(c) \leftarrow label(m.predict(X), c)$
  for m in M ▷ Model Evaluation
    for m' in $M \backslash \{m\}$
      for c in C
        $S[m, m', c] \leftarrow d(L_m(c), L_{m'}(c))$
    $\mathbb{S}[m] \leftarrow mean(S[m, \cdot, \cdot])$
  return $argmax(\mathbb{S})$

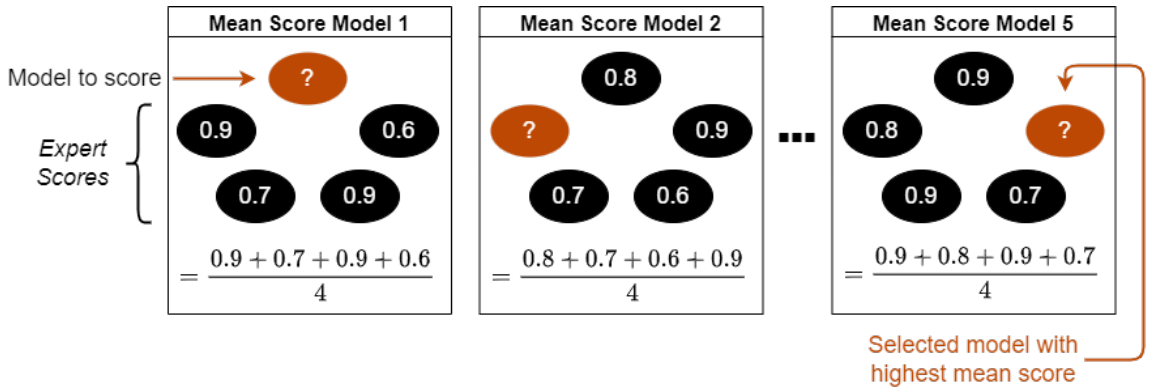


Figure 2: An instance of N-1 Experts with 5 candidate models. Using the predictions of the other expert models as ground truth, we calculate the score of the candidate model on each and aggregate over the contamination factors to compute the Expert Scores. Each boxed group represents the calculation of one mean model score, based on which N-1 Experts selects the best performing one.

B Datasets Statistics

In Table 2 and Table 3, we show the two sets of datasets that we use for our experiments.

Table 2: Statistics of the **ODDS** datasets.

Dataset name	N samples	N features	Contamination
arrhythmia	452	274	0.15
cardio	1831	21	0.10
glass	214	9	0.04
ionosphere	351	33	0.36
letter	1600	32	0.06
lympho	148	18	0.04
mnist	7603	100	0.09
musk	3062	166	0.03
optdigits	5216	64	0.03
pendigits	6870	16	0.02
satellite	6435	36	0.32
satimage-2	5803	36	0.01
vertebral	240	6	0.12
wbc	378	30	0.06

Table 3: Statistics of the **SSL** datasets.

Dataset name	N samples	N features	Contamination
ssl-1	7894	476	0.09
ssl-2	1061	436	0.18
ssl-3	15945	322	0.20
ssl-4	13719	280	0.16
ssl-5	41340	339	0.12
ssl-6	2916	251	0.18
ssl-7	2940	240	0.15
ssl-8	3560	287	0.13

C Statistics of the Sets of Models

In Table 4 we display the distribution of candidate model types in each of the **OC** and **RC** sets. In **OC**, each model type appears only once, corresponding to a single high-performance hyperparameter configuration. On the other hand **RC** contains 5 fixed configurations picked at random for each model type. Except for the AutoEncoder-based detector, which is implemented internally based on the Keras library (Chollet et al., 2015), all models are trained using the open-source implementation from the PyOD library (Zhao et al., 2019b).

Table 4: Number of hyperparameter configurations for each candidate model type.

Model Type	**OC**	**RC**
LODA (Pevný, 2016)	0	5
Isolation Forest (Liu et al., 2008)	1	5
k-Nearest Neighbors (Ramaswamy et al., 2000)	1	5
Local Outlier Factor (Breunig et al., 2000)	1	5
Histogram-Based Outlier Score (Goldstein and Dengel, 2012)	1	5
One-Class SVM (Schölkopf et al., 2001)	1	5
Outlier Detection PCA (Shyu et al., 2003)	1	0
Minimum Covariance Determinant (Hardin and Rocke, 2004)	1	0
AutoEncoder-based detector	1	0

D Individual Dataset Regrets

In Table 5 and Table 6 are displayed the mean regret and variance of the selection methods on the **SSL** and **ODDS** datasets, for both **OC** and **RC** models. Each entry is of the form $\mu \pm \sigma^2$, where μ and σ^2 are respectively the mean regret and regret variance across all 15 independent runs. For a given (dataset, set of models) pair, the smallest mean across the two selection methods is bolded if the difference is significant at a 5% significance level according to a Wilcoxon signed rank text. The smallest variance is bolded in the same manner. One can observe that N-1 Experts nearly always has a smaller regret variance than MetaOD. Furthermore, N-1 Experts's variance is always smaller among the statistically-significant differences, on both sets of datasets.

Table 5: Regret mean and variance on individual datasets for the **SSL** datasets.

Dataset	Optimized Models (**OC**)		Random Models (**RC**)	
	N-1 Experts	MetaOD	N-1 Experts	MetaOD
ssl-1	0.049 ± 0.000	0.061 ± 0.011	**0.018** ± 0.001	0.095 ± 0.034
ssl-2	0.201 ± **0.002**	**0.124** ± 0.112	0.134 ± 0.004	0.160 ± 0.008
ssl-3	0.071 ± 0.002	0.095 ± 0.003	**0.107** ± 0.003	0.144 ± 0.005
ssl-4	**0.006** ± **0.000**	0.111 ± 0.013	0.118 ± 0.011	0.091 ± 0.005
ssl-5	**0.011** ± **0.000**	0.150 ± 0.024	0.078 ± 0.007	0.118 ± 0.023
ssl-6	0.186 ± 0.016	0.273 ± 0.034	**0.101** ± 0.018	0.205 ± 0.026
ssl-7	**0.144** ± **0.010**	0.262 ± 0.035	**0.093** ± 0.011	0.167 ± 0.022
ssl-8	0.178 ± **0.003**	**0.106** ± 0.011	0.121 ± 0.029	0.212 ± 0.066

Table 6: Regret mean and variance on individual datasets for the **ODDS** datasets.

Dataset	Optimized Models (**OC**)		Random Models (**RC**)	
	N-1 Experts	MetaOD	N-1 Experts	MetaOD
arrhythmia	0.015 ± 0.000	0.026 ± 0.002	**0.017** ± **0.000**	0.107 ± 0.021
cardio	**0.025** ± **0.000**	0.084 ± 0.006	0.065 ± **0.006**	0.221 ± 0.111
glass	0.169 ± **0.000**	0.179 ± 0.003	**0.035** ± 0.001	0.152 ± 0.004
ionosphere	**0.099** ± **0.000**	0.221 ± 0.025	**0.078** ± **0.001**	0.238 ± 0.056
letter	0.284 ± 0.000	0.206 ± 0.015	0.224 ± 0.015	0.238 ± 0.015
lympho	0.013 ± 0.000	0.006 ± 0.000	0.026 ± 0.000	0.051 ± 0.015
mnist	0.057 ± **0.001**	0.103 ± 0.009	0.166 ± **0.001**	0.209 ± 0.019
musk	0.000 ± 0.000	0.076 ± 0.025	**0.003** ± **0.000**	0.382 ± 0.159
optdigits	0.184 ± 0.002	0.141 ± 0.022	0.512 ± **0.000**	**0.299** ± 0.020
pendigits	0.020 ± 0.000	0.042 ± 0.004	0.187 ± 0.002	0.123 ± 0.021
satellite	0.110 ± **0.000**	0.083 ± 0.004	0.118 ± 0.002	0.140 ± 0.006
satimage-2	0.005 ± 0.000	0.042 ± 0.012	0.070 ± 0.018	0.084 ± 0.025
vertebral	0.269 ± 0.001	0.256 ± 0.006	0.309 ± **0.001**	**0.202** ± 0.012
wbc	0.014 ± 0.000	0.012 ± 0.000	0.023 ± **0.000**	0.207 ± 0.136

E Running Times

Our proposed N-1 Experts framework does not need any labelled historical datasets, which makes it fully unsupervised. As such its offline model training and meta-learning parts are substantially faster (in fact, there is no offline training cost). However, when model selection is performed online on a new dataset, it requires 1) training multiple candidate models ("experts"), and then 2) some time to rank these models according to their outlier scores. In Table 7, we count the time required for these two steps as "online model training" and "online meta-learning", respectively. This means that while MetaOD requires substantial additional offline running time, N-1 Experts requires more online running time to select a model. Nevertheless, N-1 Experts still requires less running time overall.

Table 7: Average running times (seconds) to select a model on a single dataset, without parallelism. Model training corresponds to pre-training the candidate models; meta-learning corresponds to the time to train the meta-learning method (or score the models, in the case of N-1 Experts) in order to select a model. The total running time of a method is the sum of those four components.

Method	Model training		Meta-Learning	
	Offline	Online	Offline	Online
N-1 Experts	0.0	260.0	0.0	7.5
MetaOD	2236.6	25.2	209.8	0.1