Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 200 (2022) 984–992

Procedia Computer Science

www.elsevier.com/locate/procedia

3rd International Conference on Industry 4.0 and Smart Manufacturing

Little data is often enough for distance-based outlier detection

David Muhr[a,c], Michael Affenzeller[b,c]

[a] *BMW Group, Steyr, Austria*

[b] *Heuristic and Evolutionary Algorithms Laboratory, University of Applied Sciences Upper Austria, Hagenberg, Austria*

[c] *Institute for Formal Models and Verification, Johannes Kepler University, Linz, Austria*

Abstract

Many real-world use cases benefit from fast training and prediction times, and much research went into speeding up distance-based outlier detection methods to millions of data points. Contrary to popular belief, our findings suggest that little data is often enough for distance-based outlier detection models. We show that using only a tiny fraction of the data to train distance-based outlier detection models often leads to no significant reduction in predictive performance and detection variance over a wide range of tabular datasets. Furthermore, we compare a data reduction based on random subsampling and clustering-based prototypes and show that both approaches yield similar outlier detection results. Simple random subsampling, thus, proves to be a useful benchmark and baseline for future research on speeding up distance-based outlier detection.

Keywords: outlier detection; anomaly detection; clustering; prototypes; unsupervised; nearest neighbors; local outlier factor; knn; lof; k-means;

1. Introduction

Outlier or *anomaly* are the two terms most commonly used in the context of *outlier detection* or *anomaly detection*. An outlier is frequently defined as "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data" [5]. Outlier detection is the research area that studies the detection of such inconsistent observations. Outliers are by nature infrequent events (e.g., rare medical conditions or machine failures), and labels are often difficult to obtain. Most outlier detection algorithms, therefore, operate in an *unsupervised* setting.

E-mail address: david.muhr@bmw.com

1877-0509
10.1016/j.procs.2022.01.297

1.1. Distance-based outlier detection

A broad spectrum of techniques have been proposed for outlier detection, and many reviews exist on the topic, covering various outlier detection approaches and application scenarios; see Hodge and Austin [16], Chandola et al. [9], Pimentel et al. [25] or Wang et al. [31], for example. Because there exists no rigid definition concerning which observation is an outlier, each outlier detection algorithm relies on certain assumptions to decide what qualifies an instance to be regarded as an outlier [38]. Our focus lies on *distance-based* outlier detection approaches, which assume that normal data instances occur in *dense neighborhoods*, while anomalies occur far from their closest neighbors. A key advantage of distance-based techniques is that they are unsupervised in nature and do not make any assumptions regarding the generative distribution for the data [9]. Additionally, distance-based algorithms are considered highly transparent and explainable depending on the distance function employed [6]. Distance-based outlier detection methods are often further differentiated in terms of *local* and *global* methods, which relates to the decision if the outlierness of an instance is based on the complete (global) dataset or only on a (local) selection of instances [28].

1.2. Approximate distance-based methods

Most distance-based algorithms rely on identifying an instance's nearest neighbors to determine its outlierness, which is a computationally expensive operation. The computational complexity of a single nearest-neighbor query with Euclidean distance is $O(nd)$, where n refers to the number of examples and d to the dimensionality of the dataset [19]. A large amount of research investigates speeding up distance-based outlier detection algorithms. As mentioned previously, the computational complexity depends on both the number of instances and the dimensionality of the dataset. We investigate how the number of training instances influences the prediction result; thus, our investigation solely focuses on the instance-based complexity. There are many existing approximation techniques focusing on the instance-based complexity of nearest-neighbor searches, which we summarize below.

- *Data Subsampling* refers to using only a subset of the data to learn an outlier detection model. Zimek et al. [39] show that instance subsampling can be used to create efficient distance-based outlier detection ensembles. Aggarwal and Sathe [1] investigate the theoretical foundations for subsampling methods in unsupervised outlier detection. Song et al. [30] use subsampling ensembles for outlier detection with feature-mapped neighbors.
- *Clustering and Pruning* can be used to partition the data and discarding instances that do not influence the prediction results. Yang and Huang [36] use clustering and a pruning scheme to speed up outlier detection. Wang and Zheng [32] use a similar two-stage clustering procedure to improve nearest-neighbor outlier detection. [35] suggest a pruning strategy to accelerate the identification of the top-n outliers. Kasture and Gadge [17] propose a neighbor-based outlier detection method using k-Means clustering to prune instances near the cluster centroids. Salehi et al. [27] use a k-Means clustering approach to retain only those points that are most useful for subsequent outlier detection in data streams.
- *Approximate neighbors* describes techniques that use heuristics to determine the neighbors of an instance. Pei et al. [24] define an approximation method for distance-based outlier detection using reference points. [34] propose an approximate k-nearest neighbors variant inspired by minimum spanning trees. Schubert et al. [29] present an ensemble method that approximates local neighborhoods using an ensemble of space-filling curves. Kirner et al. [20] evaluate good and bad neighborhood approximations for distance-based outlier detection ensembles. Aumüller et al. [4] provides a comprehensive benchmark of recent approximate neighbor algorithms.
- *Prototypes* describe methods that construct a set of instances to represent the entire dataset. Hart [15] and Angiulli [3] describes prototype selection methods that consist of a subset of the training data. Mollineda et al. [23] create artificial prototypes using hierarchical clustering. Harmeling et al. [14] propose an indexing scheme to identify outlier prototypes. Recently, various prototype methods were introduced for supervised nearest-neighbor approaches, see [21], [33], [37] or [13].

Besides the approximate methods, the complexity of exact neighbor queries can be reduced using various tree-based indexing structures, as studied in [19], for example. Instead of adjusting distance-based methods to use more data points, we evaluate if using less data is a viable option to speed up existing outlier detection methods. There have been some investigations about subsampling approaches in distance-based outlier detection, but those focus mainly

on ensemble learning strategies (e.g., [39] or [1]), whereas we would like to identify the effects of subsampling on individual distance-based outlier detection models. Additionally, we compare random subsampling to cluster-based sampling using prototypes, which previous researchers have not addressed.

2. Methodology

Our goal is to investigate how distance-based outlier detection algorithms are affected by a reduction of training data, either through random subsampling or through clustering-based prototypes. More specifically, our investigation is concerned with the following three research questions (RQs).

RQ1: How does random data subsampling influence distance-based outlier detection performance?

RQ2: How does prototype-based data subsampling influence distance-based outlier detection performance?

RQ3: Does prototype-based subsampling lead to better outlier detection results than random subsampling?

In the following sections, we describe the different datasets used for evaluation (2.1), our random subsampling and prototype methodology (2.2), the algorithms used in our study (2.3), and the evaluation method used to answer the posed research questions (2.4).

2.1. Datasets

The datasets used in our study mostly stem from Campos et al.'s [8] review on the evaluation of unsupervised outlier detection and comprise a range of tabular datasets. The sets have either previously appeared in the research literature or are originally intended for classification, where one or more classes have a semantic interpretation as outliers. All semantically meaningful datasets are sampled to different outliers fractions. The outlier sampling fractions are $\{0.02, 0.05, 0.1, 0.2\}$. To mitigate the impact of randomization when downsampling, the procedure is repeated ten times for each dataset, resulting in 10 different variants for these datasets. Four of the datasets provided in [8] contain less than 200 instances, which we remove from our evaluation.

Preprocessing includes removing duplicates, the transformation of categorical attributes, and linear feature normalization to the range $[0, 1]$. Detailed information on the individual datasets and the preprocessing involved can be found in [8]. Additionally, we evaluate our results on twelve proprietary datasets consisting of high-dimensional manufacturing sensor data. The normal data for each sensor data set is sampled ten times from a large pool of data. In total, 31 datasets are included, and an overview of all datasets and their characteristics can be found in table 1.

2.2. Sampling and Prototypes

First, we randomly split each dataset into a 50% training and test set. We split the data so that the training and test set contains an approximately equal amount of outliers, also known as stratification. Let N_{train} be the number of instances in the training set and N_{test} be the number of instances in the test set. For a number of fixed proportions $p \in \{0.1, 0.2, \ldots, 0.9\}$ of N_{train}, we evaluate how a reduction to that proportion of the training data influences our outlier detection performance. The exact amount of training instances for a fixed proportion p is defined as $N_{\text{train}}^{(p)} := \lfloor p \cdot N_{\text{train}} \rfloor$.

For RQ1, we randomly subsample $N_{\text{train}}^{(p)}$ instances for each fixed proportion p. For RQ2, we calculate $N_{\text{train}}^{(p)}$ k-Means clusters for each proportion p and use the cluster centroids as prototypes for the training data. We use k-Means as our prototype implementation because of its wide use, scalable implementations, and the cluster centroids intuitively capturing the notion of a prototype.

2.3. Algorithms

Because it is unfeasible to evaluate all available distance-based outlier detection algorithms in detail, we use k-Nearest Neighbors (KNN) and Local Outlier Factor (LOF) as a proxy for other global and local outlier detection techniques. Many distance-based outlier detection algorithms follow the basic neighbor-based approaches expressed in KNN and LOF, see [28] or [2] for specific reviews on more recent distance-based methods. The benefit of using

Table 1. The datasets used for evaluation where N denotes the number of samples,$|O|$ the number of outliers, and d the dimensionality of the dataset.

Dataset	Description	N	$\|O\|$	d	Ref.
Datasets used in the literature					
ALOI	Images represented with histograms features	50,000	1508	27	[12]
Glass	A dataset describing types of glass using class 6 as outliers	214	9	7	[18]
Ionosphere	Differentiates bad radars for structures in the Ionosphere	351	126	32	[18]
KDDCup99	Data describing network intrusions or attacks	60,632	246	41	[11]
PenDigits	Hand-written digits with class 4 downsampled as outliers	9,868	20	16	[11]
Shuttle	Space shuttle data using class 2 as outliers	1,013	13	9	[11]
Waveform	Three classes of waves and with class 0 downsampled as outliers	3,443	100	21	[11]
WBC	Benign or malignant cancer types with malignant as outliers	454	10	9	[11]
WDBC	Nuclear characteristics for breast cancer with malignant outliers	367	10	30	[11]
Semantically meaningful datasets					
Annthyroid	Hypothyroidism data with classes other than normal as outliers	7,200	534	21	[11]
Arrhythmia	Cardiac arrhythmia patient records with arrhythmia as outliers	450	206	259	[11]
Cardiotocography	Data set related to heart diseases with other than normal as outliers	2,126	471	21	[11]
HeartDisease	Medical data on heart problems with affected patients are outliers	270	120	13	[11]
InternetAds	Web images classified as ads or not with ads being outliers	3,264	454	1,555	[11]
PageBlocks	Different types of blocks in document pages with non-text as outliers	5,473	560	10	[11]
Pima	Patients suffering from diabetes are considered outliers	768	268	8	[11]
SpamBase	Data set representing emails classified as normal or spam (outliers)	4,601	1,813	57	[11]
Stamps	Differentiate geniune (ink) stamps from forged stamps (outliers)	340	31	9	[22]
Wilt	Differentiates diseased trees (outliers) from other land covers	4,839	261	5	[11]
Proprietary datasets					
Sensor1a	Detect different defects (outliers) using sensor point 1 on machine 'a'	1,000	30	3001	-
Sensor1b	Detect different defects (outliers) using sensor point 1 on machine 'b'	1,000	30	3001	-
Sensor1c	Detect different defects (outliers) using sensor point 1 on machine 'c'	1,000	30	3001	-
Sensor2a	Detect different defects (outliers) using sensor point 2 on machine 'a'	1,000	30	3001	-
Sensor2b	Detect different defects (outliers) using sensor point 2 on machine 'b'	1,000	30	3001	-
Sensor2c	Detect different defects (outliers) using sensor point 2 on machine 'c'	1,000	30	3001	-
Sensor3a	Detect different defects (outliers) using sensor point 3 on machine 'a'	1,000	30	1440	-
Sensor3b	Detect different defects (outliers) using sensor point 3 on machine 'b'	1,000	30	1440	-
Sensor3c	Detect different defects (outliers) using sensor point 3 on machine 'c'	1,000	30	1440	-
Sensor4a	Detect different defects (outliers) using sensor point 4 on machine 'a'	1,000	30	1440	-
Sensor4b	Detect different defects (outliers) using sensor point 4 on machine 'b'	1,000	30	1440	-
Sensor4c	Detect different defects (outliers) using sensor point 4 on machine 'c'	1,000	30	1440	-

KNN and LOF for evaluation is that they depend on a single parameter k once the distance metric is fixed. We use the Euclidean distance metric for our evaluations. For each proportion p of the training data, we optimize the hyperparameter k individually for both random subsampling and the cluster prototypes. We assume that k lies in $\{1, 2, \ldots, 10\} \cup \{15, 20, \ldots, 100\}$, but restrict k to lie below $N_{\text{train}}^{(p)} - 1$. For the cluster prototypes, we use a fast k-Means approach proposed by Ding et al. [10]. Contrary to the traditional clustering objective, we are not looking to find the cluster substructure in the data. Instead, we attempt to combine similar points and keep only one point (the centroid) as a prototype of each cluster.

2.3.1. k-Nearest Neighbors

Ramaswamy et al.[26] propose to use an instance's distance to its k^{th}-nearest neighbor to determine its outlier score. Specifically, for a number of neighbors k and an instance x, let $D^{(k)}(x)$ denote the distance of the k^{th} nearest neighbor of x. $D^{(k)}(x)$ can intuitively be seen as a measure of how much of an outlier an instance x is. Larger values of $D^{(k)}(x)$ imply sparser neighborhoods and are thus typically stronger outliers than points belonging to dense neighborhoods with lower values of $D^{(k)}(x)$. This outlier detection approach is typically referred to as k-nearest neighbors outlier

detection. To decide if an instance is an outlier or not, the value of $D^{(k)}(x)$ is compared to the k^{th} neighbor distances of the rest of the dataset. Because the distance is compared to the rest of the dataset, KNN is seen as a global method.

2.3.2. Local Outlier Factor

Breunig et al.[7] first introduce the concept of locality in distance-based outlier detection with the local outlier factor. LOF compares the density of each instance x with the density of the k-nearest neighbors of x. A score of approximately 1 indicates that the corresponding object is located within a region of homogeneous density. If the difference between the density in the local neighborhood of x, and the density around the neighbors of x is higher, x gets assigned a higher LOF value. LOF is considered a local method because the outlierness of an instance depends on how isolated an instance is in relation to its surrounding neighborhood.

2.4. Evaluation

The most popular evaluation measure in unsupervised outlier detection is based on the Receiver Operating Characteristic (ROC). A ROC can be summarized by a single value known as the area under the ROC curve (ROC AUC). A perfect ranking would result in a ROC AUC value of 1, whereas an inverted perfect ranking would result in a value of 0. A value of 0.5 can be seen as random guessing [8]. For each k, we evaluate the ROC AUC score for KNN and LOF using the whole training set, and each training set proportion p. We report only the best result achieved with a specific value for k. If the same evaluation score is achieved for multiple values of k, we report the smallest k.

3. Results

In this section, we show that distance-based outlier detection methods are robust against data sampling with regards to detection performance (3.1) and detection variance (3.2). We further show that k-Means cluster prototypes can result in better predictive performance and lower variance when compared to random subsampling (3.3) in global outlier detection with KNN, but not in local outlier detection with LOF. In addition to the aggregated results shown in this paper, we provide detailed per-dataset results including the optimized hyper-parameters online[1].

3.1. Effect on detection performance

Averaged over all datasets, the detection performance decreases slightly with a reduction of training data as visible in Figure 1. Randomly sampling 10% of the training data resulted in a 1.14% ± 5.33% ROC AUC reduction for KNN and 1.81% ± 5.04% reduction for LOF. Using 10% prototypes instead, the mean reduction is 0.33% ± 5.07% and 1.92% ± 5.20% respectively. Thus, over all datasets, LOF is slightly more sensitive to data reduction compared to KNN. This stability of KNN over LOF has also been observed in [1]. Interestingly, a data reduction to 10% of the training data leads to no score decrease or a score improvement in ~ 38% of the datasets with random sampling and ~ 42% of the datasets with prototype sampling with KNN and ~ 28% for both sampling approaches with LOF.

There are cases where a data reduction leads to better ROC AUC scores when the optimal amount of neighbors falls above our hyperparameter search range (100 neighbors). On the other hand, if the amount of neighbors is already low for the full training set, a reduction in data typically leads to a decline in performance. These facts have also been observed by [1]. Note that optimizing k for each data partition is essential when using random or prototype-based sampling; otherwise, an incorrect bias for a specific data set size is introduced, as exemplified by the delusory results in [39], where the authors observed that a reduction in data leads to consistently better ensemble models.

3.2. Effect on detection variance

Ten randomly sampled versions exist for all subsampled and proprietary datasets, which we use to determine the variance of the achieved detection results. Perhaps surprisingly, the difference in variation across sampling and prototype fractions is low for both KNN and LOF, see Figure 2.

[1] https://davnn.github.io/little-data/

Fig. 1. ROC AUC scores averaged over all datasets and the corresponding standard deviation.

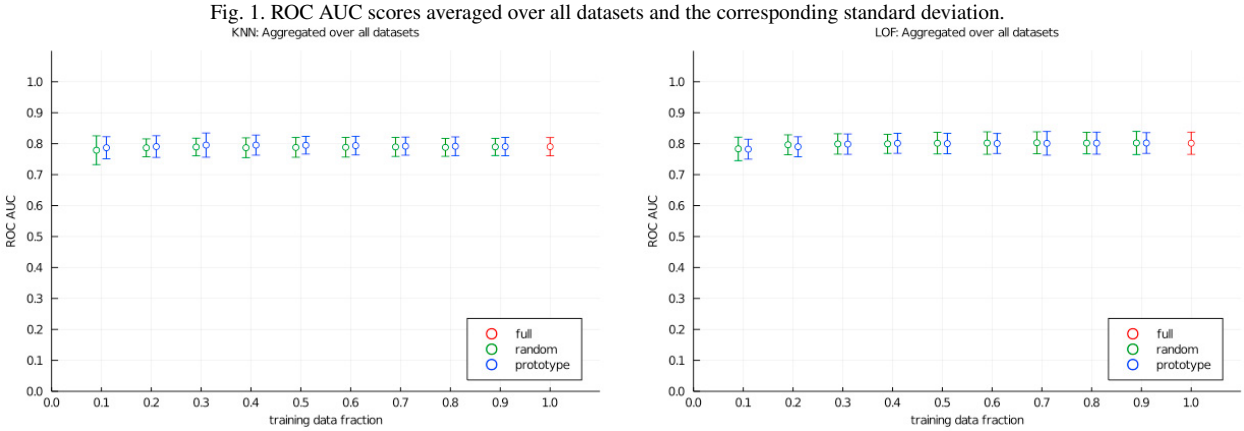
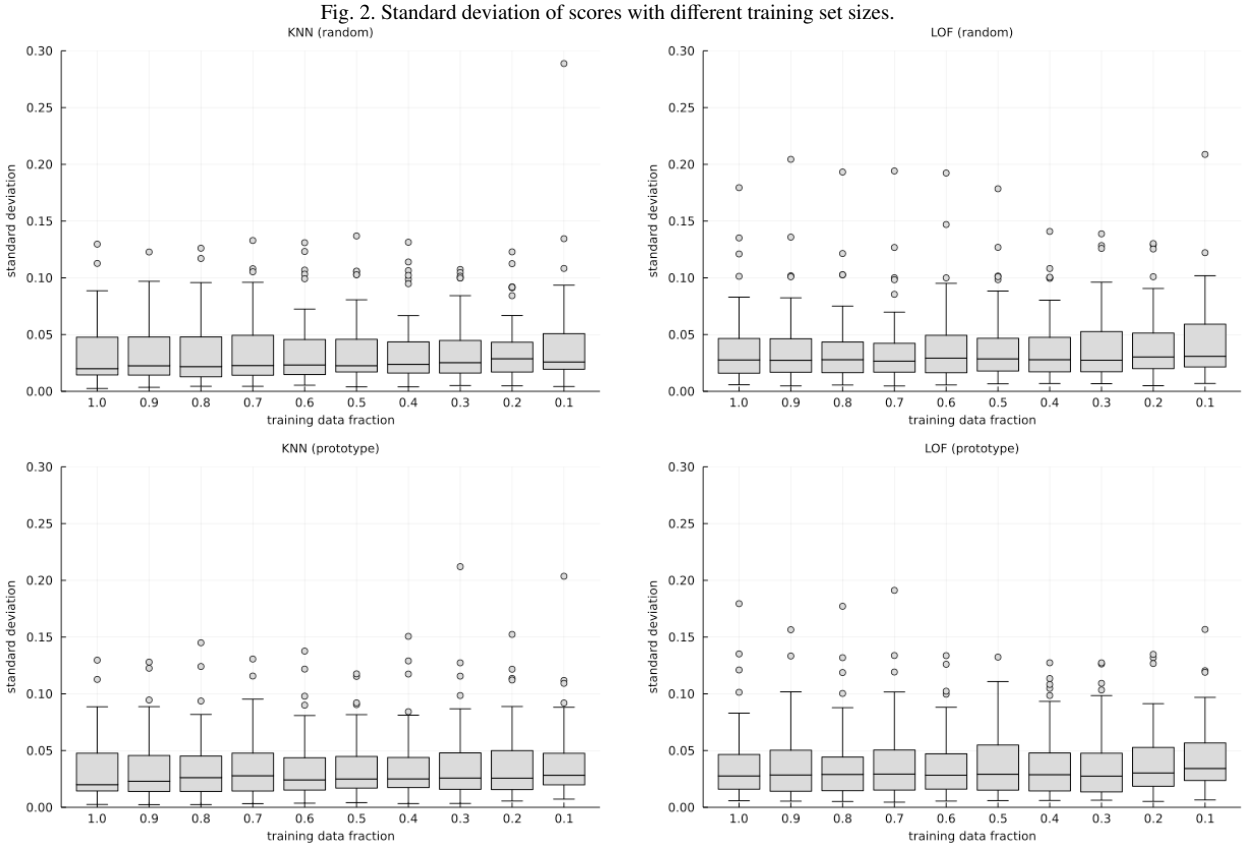


Fig. 2. Standard deviation of scores with different training set sizes.

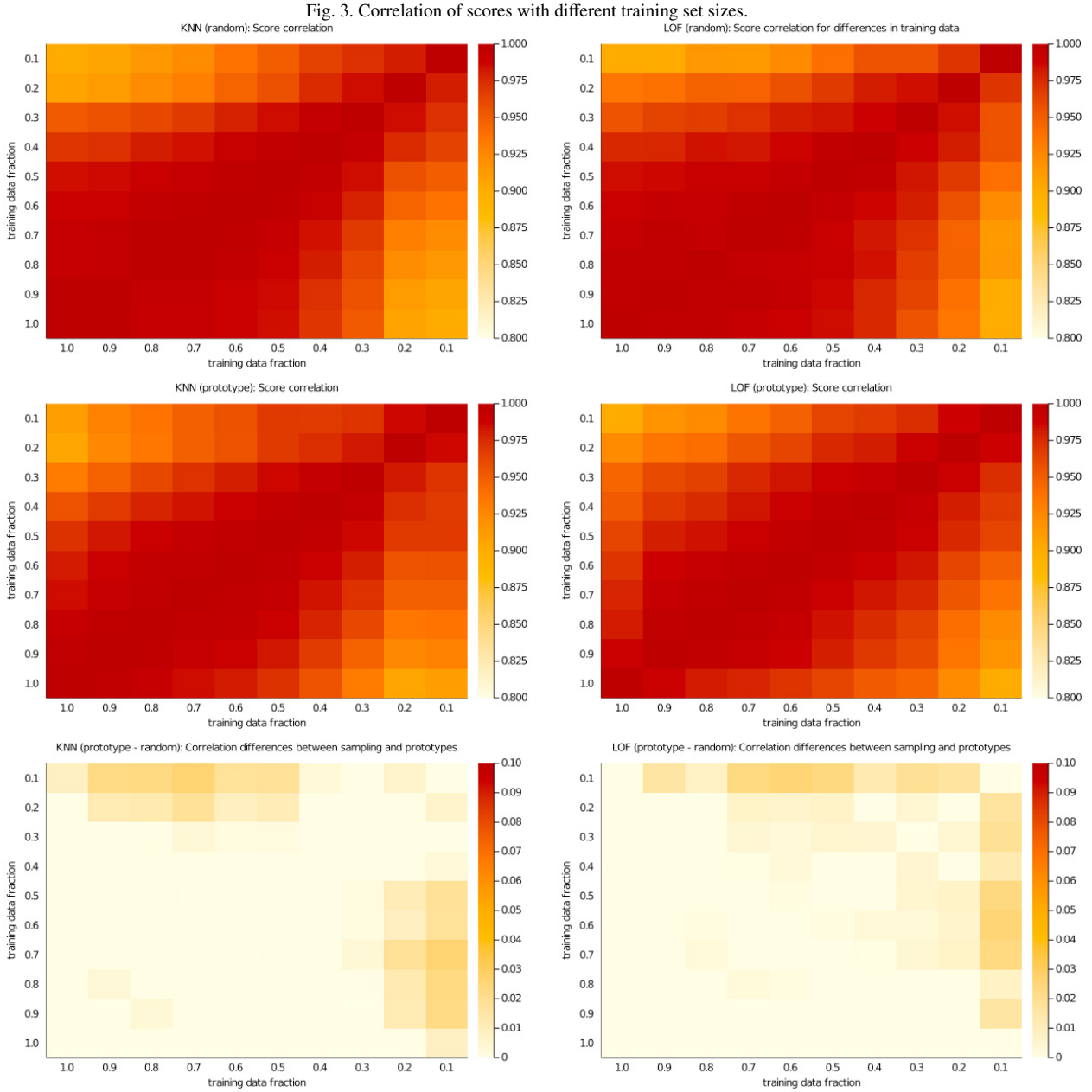


Using the five datasets containing different outlier fractions from 2% to 20%, we additionally analyze how the outlier fractions influence the detection variance. A consistent pattern found is that the detection variance decreases with a higher outlier fraction for both KNN and LOF and over all values of p. The average variance linearly decreases from 8.79% ± 4.83% (KNN) and 9.15% ± 4.93% (LOF) at 2% outliers to 3.46% ± 2.32% and 3.71% ± 2.53% at 20% outliers with no significant differences between random subsampling and prototypes. An additional visualization detailing the relationship between the variation of scores and different outlier fractions can be found online[1].

3.3. Comparison of random sampling and prototypes

The correlation between different ROC AUC scores is above 90% for all sampling sizes for both KNN and LOF. Using prototypes instead of random sampling leads to better correlation when the sampling size is small, as visible in Figure 3. Furthermore, we use a Wilcoxon signed-rank test to compare the random and prototype ROC AUC scores statistically. Over all sample sizes, there is a small but significant ($p < 0.01$) improvement using prototype sampling over random sampling for KNN. For LOF, there is no significant difference visible ($p \approx 0.18$). The difference in detection variance between random sampling and prototypes shows a similar picture. Prototypes lead to significantly lower detection variance for KNN ($p \approx 0.03$), but not for LOF ($p \approx 0.23$).

Fig. 3. Correlation of scores with different training set sizes.



In summary, we find that both KNN and LOF are highly robust against both random- and prototype-based data sampling (RQ1, RQ2). Prototype-based sampling leads to a better score correlation compared to random sampling for both KNN and LOF, but only for KNN, the ROC AUC score improvements are significant (RQ3).

4. Conclusion

The trade-off between bias and variance is one of the central machine learning challenges. Using more data is the most trivial approach to reduce the variance of an estimator. However, there is also a trade-off between learning- and prediction-speed and predictive performance, which is especially relevant in unsupervised learning, where vast amounts of data might be available. We show that simple random data subsampling and prototype-based data reduction are valuable strategies to accelerate learning and prediction with the analyzed distance-based outlier detection approaches. In cases where an impractically large number of neighbors best describes the outlier density, a data reduction is particularly suited to improve prediction speed and predictive performance. Current approximation techniques to accelerate distance-based outlier detection are typically evaluated using the complete training data set as a benchmark for speed and performance. Developers of future speed-up methods for distance-based outlier detection should additionally include random subsampling as a simple benchmark and baseline. Unfortunately, the datasets currently used in the outlier detection community are often very small and do not capture the challenges inherent in real-world use cases. Thus, our study uses datasets that might not represent real-world outlier detection challenges, but we tried to mitigate this by including proprietary datasets from real-world use cases, and we plan to publish these datasets in the future. Another drawback in our study is the simple k-Means-based prototype strategy, which showed mixed results overall. In many use cases, prototypes did not show significant improvements over random subsampling, but some use cases showed promising results. More insights are necessary to understand under which circumstances prototypes lead to better outlier detection results, which provides an exciting research area for future studies. In conclusion, our findings raise intriguing questions about data sampling in distance-based outlier detection. In future investigations regarding the acceleration of distance-based outlier detection, researchers should shed more light on the properties needed for data sampling to be effective by using it as a simple baseline for novel methods.

References

[1] Aggarwal, C.C., Sathe, S., 2015. Theoretical foundations and algorithms for outlier ensembles. ACM SIGKDD Explorations Newsletter 17, 24–47. doi:10.1145/2830544.2830549.

[2] Alghushairy, O., Alsini, R., Soule, T., Ma, X., 2021. A review of local outlier factor algorithms for outlier detection in big data streams. Big Data and Cognitive Computing 5, 1. doi:10.3390/bdcc5010001.

[3] Angiulli, F., 2005. Fast condensed nearest neighbor rule, in: Dzeroski, S. (Ed.), Proceedings of the 22nd international conference on Machine learning - ICML '05, ACM Press, New York, New York, USA. pp. 25–32. doi:10.1145/1102351.1102355.

[4] Aumüller, M., Bernhardsson, E., Faithfull, A., 2020. Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. Information Systems 87, 101374. doi:10.1016/j.is.2019.02.006.

[5] Barnett, V., Lewis, T., 1978. Outliers in Statistical Data. John Wiley & Sons, Inc.

[6] Belle, V., Papantonis, I., 2021. Principles and Practice of Explainable Machine Learning. Frontiers in Big Data 4, 688969. URL: https://www.frontiersin.org/articles/10.3389/fdata.2021.688969/full, doi:10.3389/fdata.2021.688969.

[7] Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J., 2000. Lof: Identifying density-based local outliers, in: Dunham, M., Naughton, J.F., Chen, W., Koudas, N. (Eds.), Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Association for Computing Machinery, New York, NY, USA. pp. 93–104. doi:10.1145/342009.335388.

[8] Campos, G.O., Zimek, A., Sander, J., Campello, R.J.G.B., Micenková, B., Schubert, E., Assent, I., Houle, M.E., 2016. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Mining and Knowledge Discovery 30, 891–927. doi:10.1007/s10618-015-0444-8.

[9] Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection. ACM Computing Surveys 41, 1–58. doi:10.1145/1541880.1541882.

[10] Ding, Y., Zhao, Y., Shen, X., Musuvathi, M., Mytkowicz, T., 2015. Yinyang k-means: A drop-in replacement of the classic k-means with consistent speedup, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, JMLR.org. pp. 579–587.

[11] Dua, D., Graff, C., 2017. Uci machine learning repository. URL: http://archive.ics.uci.edu/ml.

[12] Geusebroek, J.M., Burghouts, G.J., Smeulders, A.W., 2005. The amsterdam library of object images. International Journal of Computer Vision 61, 103–112. URL: https://link.springer.com/article/10.1023/B:VISI.0000042993.50813.60, doi:10.1023/B:VISI.0000042993.50813.60.

[13] Gupta, C., Suggala, A.S., Goyal, A., Simhadri, H.V., Paranjape, B., Kumar, A., Goyal, S., Udupa, R., Varma, M., Jain, P., 2017. Protonn: Compressed and accurate knn for resource-scarce devices, in: Proceedings of the 34th International Conference on Machine Learning - Volume 70, JMLR.org. pp. 1331–1340. doi:10.5555/3305381.3305519.

[14] Harmeling, S., Dornhege, G., Tax, D., Meinecke, F., Müller, K.R., 2006. From outliers to prototypes: Ordering data. Neurocomputing 69, 1608–1618. doi:10.1016/j.neucom.2005.05.015.

[15] Hart, P., 1968. The condensed nearest neighbor rule (corresp.). IEEE Trans. Inf. Theor. 14, 515–516. doi:10.1109/TIT.1968.1054155.

[16] Hodge, V., Austin, J., 2004. A survey of outlier detection methodologies. Artificial Intelligence Review 22, 85–126. doi:10.1023/B:AIRE.0000045502.10941.a9.

[17] Kasture, P., Gadge, J., 2012. Cluster based outlier detection. International Journal of Computer Applications 58, 11–15. doi:10.5120/9317-3549.

[18] Keller, F., Muller, E., Bohm, K., 2012. Hics: High contrast subspaces for density-based outlier ranking, in: 2012 IEEE 28th International Conference on Data Engineering, pp. 1037–1048. doi:10.1109/ICDE.2012.88.

[19] Kibriya, A.M., Frank, E., 2007. An empirical comparison of exact nearest neighbour algorithms, in: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Springer-Verlag, Berlin, Heidelberg. pp. 140–151. doi:10.1007/978-3-540-74976-9{\textunderscore}16.

[20] Kirner, E., Schubert, E., Zimek, A., 2017. Good and bad neighborhood approximations for outlier detection ensembles. Lecture Notes in Computer Science 10609, 173–187. doi:10.1007/978-3-319-68474-1{\textunderscore}12.

[21] Kusner, M.J., Tyree, S., Weinberger, K., Agrawal, K., 2014. Stochastic neighbor compression, in: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, JMLR.org. pp. II–622–II–630.

[22] Micenková, B., van Beusekom, J., Shafait, F., 2015. Stamp verification for automated document authentication, in: Garain, U., Shafait, F. (Eds.), Computational forensics, Springer, Cham. pp. 117–129. doi:10.1007/978-3-319-20125-2{\textunderscore}11.

[23] Mollineda, R.A., Ferri, F.J., Vidal, E., 2002. An efficient prototype merging strategy for the condensed 1-nn rule through class-conditional hierarchical clustering. Pattern Recognition 35, 2771–2782. URL: https://www.sciencedirect.com/science/article/pii/S0031320301002084, doi:10.1016/S0031-3203(01)00208-4.

[24] Pei, Y., Zaiane, O., Gao, Y., 2006. An efficient reference-based approach to outlier detection in large datasets, in: Sixth International Conference on Data Mining (ICDM'06), IEEE. pp. 478–487. doi:10.1109/ICDM.2006.17.

[25] Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L., 2014. A review of novelty detection. Signal Processing 99, 215–249. doi:10.1016/j.sigpro.2013.12.026.

[26] Ramaswamy, S., Rastogi, R., Shim, K., 2000. Efficient algorithms for mining outliers from large data sets. SIGMOD Rec 29, 427–438. doi:10.1145/335191.335437.

[27] Salehi, M., Leckie, C., Bezdek, J.C., Vaithianathan, T., Zhang, X., 2016. Fast memory efficient local outlier detection in data streams. IEEE Transactions on Knowledge and Data Engineering 28, 3246–3260. doi:10.1109/TKDE.2016.2597833.

[28] Schubert, E., Zimek, A., Kriegel, H.P., 2014. Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. Data Mining and Knowledge Discovery 28, 190–237. doi:10.1007/s10618-012-0300-z.

[29] Schubert, E., Zimek, A., Kriegel, H.P., 2015. Fast and scalable outlier detection with approximate nearest neighbor ensembles, in: Renz, M., Shahabi, C., Zhou, X., Cheema, M.A. (Eds.), Database systems for advanced applications, Springer, Cham. pp. 19–36.

[30] Song, H., Jiang, Z., Men, A., Yang, B., 2017. A hybrid semi-supervised anomaly detection model for high-dimensional data. Computational intelligence and neuroscience 2017, 8501683. doi:10.1155/2017/8501683.

[31] Wang, H., Bah, M.J., Hammad, M., 2019. Progress in outlier detection techniques: A survey. IEEE Access 7, 107964–108000. doi:10.1109/ACCESS.2019.2932769.

[32] Wang, Q., Zheng, M., 2010. An improved knn based outlier detection algorithm for large datasets, in: Cao, L., Feng, Y., Zhong, J. (Eds.), Advanced Data Mining and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg. volume 6440 of *Lecture Notes in Computer Science*, pp. 585–592. doi:10.1007/978-3-642-17316-5{\textunderscore}56.

[33] Wang, W., Chen, C., Chen, W., Rai, P., Carin, L., 2016. Deep metric learning with data summarization, in: European Conference on Machine Learning and Knowledge Discovery in Databases - Volume 9851, Springer-Verlag, Berlin, Heidelberg. pp. 777–794. doi:10.1007/978-3-319-46128-1{\textunderscore}49.

[34] Wang, X., Wang, X.L., Ma, Y., Wilkes, D.M., 2015. A fast mst-inspired knn-based outlier detection method. Information Systems 48, 89–112. URL: https://www.sciencedirect.com/science/article/pii/S0306437914001331, doi:10.1016/j.is.2014.09.002.

[35] Yan, Y., Cao, L., Rundensteiner, E.A., 2017. Scalable top-n local outlier detection, in: Matwin, S., Yu, S., Farooq, F. (Eds.), Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA. pp. 1235–1244. doi:10.1145/3097983.3098191.

[36] Yang, P., Huang, B., 2008. Knn based outlier detection algorithm in large dataset, in: 2008 International Workshop on Education Technology and Training & 2008 International Workshop on Geoscience and Remote Sensing, IEEE. pp. 611–613. doi:10.1109/ETTandGRS.2008.306.

[37] Zhong, K., Guo, R., Kumar, S., Yan, B., Simcha, D., Dhillon Inderjit, 2017. Fast classification with binary prototypes, in: Aarti Singh, Jerry Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, PMLR, Fort Lauderdale, FL, USA. pp. 1255–1263. URL: http://proceedings.mlr.press/v54/zhong17a.html.

[38] Zimek, A., Campello, R.J., Sander, J., 2014. Ensembles for unsupervised outlier detection. ACM SIGKDD Explorations Newsletter 15, 11–22. doi:10.1145/2594473.2594476.

[39] Zimek, A., Gaudet, M., Campello, R.J., Sander, J., 2013. Subsampling for efficient and effective unsupervised outlier detection ensembles, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA. pp. 428–436. doi:10.1145/2487575.2487676.