

CHRISTIAN-ALBRECHTS-UNIVERSITÄT
ZU KIEL

DOCTORAL THESIS

**Anomaly Detection Towards Modelling with
Outlier Robustness**

Author:

Andreas LOHRER

from

Weiden i. d. Opf.

2024

*A thesis submitted in fulfillment of the requirements
for the degree of Dr. rer. nat.*

in the

Technische Fakultät

Supervisor/1st Examiner: Prof. Dr. Peer KRÖGER

2nd Examiner: Prof. Dr. Sven TOMFORDE

Tag der Mündlichen Prüfung: 04.12.2024

Declaration of Authorship

Ich, Andreas LOHRER, erkläre, dass diese Arbeit mit dem Titel, "Anomaly Detection Towards Modelling with Outlier Robustness" und die darin dargestellte Arbeit meine eigene ist. Ich bestätige, dass

- die Abhandlung, abgesehen von der Beratung durch die Betreuerin oder den Betreuer nach Inhalt und Form meine eigene Arbeit ist.
- die Arbeit noch nie ganz oder zum Teil schon einer anderen Stelle im Rahmen eines Prüfungsverfahrens vorgelegen hat, veröffentlicht worden ist oder zur Veröffentlichung eingereicht wurde, außer die Teile dieser Dissertation, welche vorherigen Publikationen entsprechen. Für diese ist mein Eigenanteil in dieser Arbeit angegeben.
- dass die Arbeit unter Einhaltung der Regeln guter wissenschaftlicher Praxis der Deutschen Forschungsgemeinschaft entstanden ist;
- mir kein akademischer Grad entzogen wurde.

Unterschrift Doktorand:

Andreas LOHRER

Datum: Kiel, December 12, 2024

Abstract

Andreas LOHRER

Anomaly Detection Towards Modelling with Outlier Robustness

Machine learning algorithms have evolved in numerous fields into an important enabler for today's society. Thereby, anomaly detection is one of the machine learning tasks which contributes to many application domains. Anomalies, also known as outliers, can thereby be of advantageous but also of detrimental effect, so that outlier robustness is one of the desired properties when it comes to the design of performant machine learning algorithms. This dissertation elaborates on how a point anomaly detection algorithm needs to be designed to neglect negative impacts of anomalies during the learning phase in order to improve their capability to detect anomalies. Furthermore, it shows how an anytime approach allows to provide permanent response and performance improvements simultaneously for point anomaly detection. Moreover, a benchmark framework for reproducible point anomaly detection is proposed. In addition to that, traditional and deep learning approaches for group anomaly detection on mobility data are introduced, as well as a deep 3D convolutional neural network for robust soil parameter prediction on hyperspectral satellite image data.

Abstract

Andreas LOHRER

Anomaly Detection Towards Modelling with Outlier Robustness

Maschinelle Lernverfahren haben sich auf zahlreichen Gebieten zu wichtigen Wegbereitern für die heutige Gesellschaft entwickelt. Dabei ist Anomalieerkennung eine der Aufgaben des maschinellen Lernens, welche einen enormen Beitrag zu vielen Anwendungsdomänen liefert. Anomalien oder auch Outlier genannt können dabei sowohl einen vorteilhaften aber auch nachteiligen Charakter besitzen, sodass die Robustheit gegenüber Outliern eine der erwünschten Eigenschaften verkörpert, wenn es um die Entwicklung performanterer maschineller Lernverfahren geht. Diese Dissertation geht u.a. näher darauf ein, wie ein Punktanomalieerkennungsalgorithmus konzipiert sein muss, um negative Einflüsse von Anomalien während der Lernphase zu vernachlässigen, sodass die Fähigkeit Anomalien zu erkennen verbessert wird. Des Weiteren wird gezeigt, wie ein Anytime-Ansatz einen Punktanomalieerkennungsalgorithmus permanent rückmelden lassen und gleichzeitig dessen Leistungsfähigkeit erhöhen kann. Außerdem wird ein Benchmark-Framework zur reproduzierbaren Punktanomalieerkennung vorgestellt. Darüber hinaus werden traditionelle und tiefe neuronale Ansätze zur Gruppenanomalieerkennung auf Mobilitätsdaten eingeführt sowie ein tiefes 3D-faltiges neuronales Netzwerk zur robusten Bodenwertsvorhersage basierend auf hyperspektralen Satelliten-Bilddaten.

Acknowledgements

First of all, I would like to thank my doctoral supervisor Prof. Dr. Peer Kröger for making this PhD study possible, for giving me the freedom to shape my own research and for providing assistance whenever I asked for it. Moreover, many thanks to Prof. Dr. Sven Tomforde for reviewing this dissertation, as well as to the examination committee. Furthermore, I say thank you to my colleagues at the LMU and the CAU for the great collaboration in research and teaching: Daniyal, Claudius, Max, Mirjam, Sweetie, Nanni, Andrea, Rajat and Yunpu. In addition to that, I would like to thank my co-authors Anna Beer, Darpan Malik, Johannes Binder and Jan Deller. Last but not least, I would like to express a special thanks to my wife Julia and my daughter Hannah, my brother Hubert with family and to my mother and deceased father for all their continuous support and for strengthening my back whenever necessary.

Contents

Declaration of Authorship	iii
Abstract	v
Acknowledgements	ix
1 Introduction	1
1.1 Anomalies	2
1.2 Anomaly Detection	3
2 Research Contributions	7
2.1 Research Questions, Contributions and Thesis Structure	7
2.2 Point Anomaly Detection	10
2.2.1 CoMAD PCA-based Point Anomaly Detection	10
2.2.2 Cluster-based Anytime Point Anomaly Detection	11
2.3 Group Anomaly Detection	12
2.3.1 Deep Group Anomaly Detection	12
2.3.2 OCSVM-based Group Anomaly Detection	13
2.4 Evaluation	14
2.5 Outlier Robustness and Applications	16
3 CoMadOut	19
4 AnyCORE	69
5 GADformer	83
6 CovEncoding	93
7 OAB	99
8 SpectralNet3D	111
9 Conclusion	117
Bibliography	119

Chapter 1

Introduction

In times where digitalization establishes no process without collecting and processing related process information on information systems, the amount of stored data and, as a result, the demand for algorithms capable of analyzing it has significantly increased. This demand is addressed by the research domain of Machine Learning, whose algorithms can meanwhile be considered as an integral part of our daily life. Thereby, the application domains and use cases can be quite versatile and represent various sectors, e.g. finance, communication, mobility, manufacturing, social media, health but also research fields such as physics, biology or geology to just name a few. Dependent on the available data and expert knowledge in these fields, algorithms are able to fulfill specific tasks in different learning settings. Tasks like e.g. the classification of patterns by category, recommendation of sets like shopping favors, prediction of parameters like weather conditions, the clustering of patterns into groups or anomaly detection are meanwhile typically known to be conducted by machine learning algorithms. Therefore, the algorithms learn either based on information they obtain from humans as so-called ground truth values, to which they learn to map related patterns, or based on a design specific pattern expectation, for which they look for in the provided data. Thereby, one distinguishes between three main learning settings based on the ground truth availability, which are 1) Unsupervised Learning, which means there is no ground truth available at all and algorithms search only design-specific for expected patterns, 2) Semisupervised Learning, which makes only use of ground truth information which is considered to be normal¹ by humans, and 3) Supervised Learning, which relies on a full availability of ground truth information for the whole data.

Since ground truth information is often difficult to obtain due to a lack of domain experts, knowledge, budget or other required resources, this thesis mainly focuses on contributing to the development and application of anomaly detection algorithms in domains with un- and semisupervised learning settings. Therefore, chapter 2.1 provides an overview of each subfield of the conducted research.

¹normal means in this context that the pattern considered as normal represents typical patterns, which appear frequently in the related application domain.

1.1 Anomalies

Since in nature, most of the data is gaussian distributed and the central limit theorem [18] states that, with a sufficient amount of samples, each distribution follows a gaussian distribution, one can see these assumptions as an initial starting point for the design of non-supervised anomaly detection algorithms. As Figure 1.1 illustrates, the majority of the data is considered as normal, whereas the minority in the tails represents outliers. Hawkins defines an outlier as follows:

«An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.». [23]

Aggarwal is even more precise in his work [1] by distinguishing between weak outliers, which he considers as noise, and strong outliers, which he considers as actual anomalies.

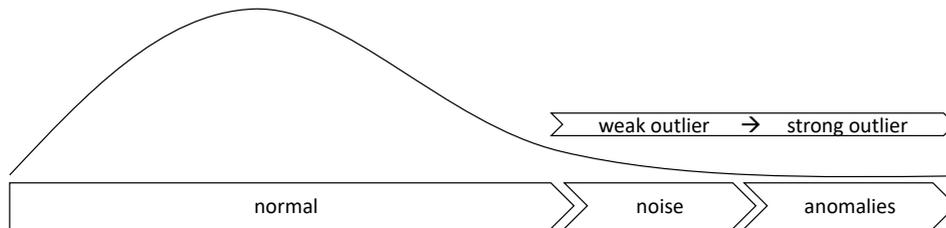


FIGURE 1.1: A one-sided gaussian tail flow from normal to abnormal instances based on [1].

Anomalies can be both beneficial and distortive at the same time. They are beneficial when they provide new insights into a domain but they can also show distortive characteristics during algorithm design, e.g. when they cause undesired rotations or translations, mean movements or provide as noise and outliers wrong normality patterns in unsupervised learning settings. In these cases, algorithms which proof a robustness towards outliers are desirable.

Besides the previously described point anomalies, there exist also so-called group anomalies or collective anomalies².

As the term suggests, group anomalies do not represent the abnormality of individual points. Instead, abnormal behavior is characterized by a group of individual points, which do not necessarily need to be abnormal themselves. [11, p. 8]

Both types, point anomalies and group anomalies, can be divided into numerous subtypes, which Foorthuis describes in his work [19, p.11] by 9 categories in total, 6 for point anomalies (I-VI) and 3 for group anomalies (VII-IX) which allow for unique distinctions. (cf. Figure 1.2)

For the contributions of this thesis, three Foorthuis anomaly subtypes are relevant, which are Type IV-a), a multivariate peripheral point anomaly, Type

²In this thesis, the terms group anomaly and collective anomaly are used interchangeably.

VII-j), a distribution-based group anomaly, and Type IX-d-i), a deviant sequence group anomaly. For an illustration of the characteristics of each of these subtypes, see Figure 1.3.

Whereas the definition of a point anomaly meets that of [23] there are also more sophisticated patterns which define anomalies, too. More precisely, they describe so-called group anomalies which are based on distributions and sequences.

Toth et al. [45] state that *"a distribution-based group anomaly is where a collection of points differs from expected group patterns"*, whereas Xiong et al. [48] describe that *"individual data points are normal, but their distribution as a group is unusual"*. According to [45, p.9] the characteristics of a distribution-based group anomaly can be described by statistical properties like location, scale, skewness, kurtosis and dependence.

The third subtype IX-d-i) is defined by [19] as a sequence-based anomaly, which represents abnormal changes in a sequence like these of amplitudes, trends, shifts, shape, variation or isolation.

These three anomaly subtypes describe the characteristics according to which the proposed approaches described in the following Chapter 1.2 learn the patterns of normality and abnormality and provide related anomaly scores.

1.2 Anomaly Detection

After the comprehensive description of the objects of interest in chapter 1.1, this chapter describes the machine learning task anomaly detection. It is used for many real world problems like fraud detection, intrusion detection, production monitoring, sky and earth observation and many more. Since the availability of ground truth information tends to cover rather normal than abnormal data for these problems, the preferred learning settings for anomaly detection are an unsupervised and semi-supervised learning setting. Due to their assumption of the majority of normal data, they allow a training without labeled anomalies.

Point Anomaly Detection Point anomaly detection (PAD) is the category which is most well known. It scores the abnormality of individual points based on various criteria. These can be the distances to principle components of PCA-based methods (e.g. [33]), but also neighborhood density of a point in relation to that of its neighbors (e.g. [7]). In addition to that, cluster memberships with point distances to their centroids allow a cluster-based anomaly detection (e.g. [32]), whereas tree-based methods (e.g. [30]) score points by their leaf depth in the tree. Possible point anomaly detection algorithm categories imply One-Class Classification (OCSVM [42]), Scoring by Reconstruction (AE [1], VAE [27], PCA [43]), Scoring by relative neighborhood densities (LOF [7], kNN [3]), Scoring by Discriminators (GAN [2]) or by tree-depth (IsolationForest [30]) to just name a few. PAD methods can be divided into unsupervised and semi-supervised methods. Unsupervised methods

include methods like e.g. LOF, IsolationForest, OCSVM, CoMadOut, AnyCORE, etc. Semi-supervised methods include OneClass-RandomForest [6], DeepSAD [40], GAN-based methods like GANomaly [2], etc. A broader overview can be found in the surveys [51], [6] and [11].

Group Anomaly Detection Group Anomaly Detection (GAD) is described by several definitions in literature. Chalapathy describes GAD *"aims to identify groups that deviate from the regular group pattern."* [10] whereas Toth and Chawla define GAD as *"the process of identifying groups that are not consistent with regular group patterns."* [45] Thereby, groups can be composed of abnormal points only, which would be so-called micro clusters, which are then detected by point-based GAD methods. Another approach called distribution-based GAD allows to detect groups which consist of individual points, no matter if they are normal or abnormal. This approach considers the abnormality of a group based on the composition and statistical properties of its partially or totally normal group members. Possible group anomaly detection algorithm categories include deep learning based methods (GADVAE [10], AAAlpha [28], GAE [12], GADformer [34]), mixture model based methods (LDA [5], MGMM [48], FGM [47], GLAD [49], CHGM [44]), SVM-based methods (OCSMM [37], SMDD [21], CovEncoding [31]) and statistical methods (FGSS [36], ERACD [13]). GAD methods can be divided into unsupervised and semi-supervised methods as well. Unsupervised methods include GADformer, OCSMM, CovEncoding, AAAlpha, etc. Semi-supervised methods include GADformer. A broader overview can be found in the survey [45].

Since this thesis mainly aims to develop new machine learning algorithms for the task of anomaly detection, the previous sections provided the required background knowledge and categorized each approach in a specific subcategory of algorithms. Therefore, one first distinguishes between approaches for point anomaly detection (cf. Chapter 2.2 for PAD) and group anomaly detection (cf. Chapter 2.3 for GAD), whereas both address the detection of the different anomaly types described in Chapter 1.1. Secondly, these two main categories are divided into further subcategories, to which the contributed approaches get related and compared to. Research Questions RQ1 to RQ6 (cf. Section 2.1) guide the reader additionally to the research core of each of the six thesis papers.

Types of Data		Anomaly Level	
		Atomic	Aggregate
Cardinality of Relationship	Univariate	<p>Quantitative attributes</p> <p>Type I: Uncommon number anomaly</p> <p>o o o o o o o o a) Extreme tail value o o o o o o b) Isolated intermediate value</p> <p>Type II: Uncommon class anomaly</p> <p>o o o o o o o o a) Unusual class o o o o o o o o b) Deviant repeater</p> <p style="text-align: center;"><i>Atomic univariate anomaly</i></p> <p>Qualitative attributes</p> <p>Type III: Simple mixed data anomaly</p> <p>o o o o o o o o a) Extreme tail uncommon class o o o o o o b) Intermediate uncommon class</p> <p>Mixed attributes</p> <p>Type III: Simple mixed data anomaly</p> <p>o o o o o o o o a) Extreme tail uncommon class o o o o o o b) Intermediate uncommon class</p>	<p>Quantitative attributes</p> <p>Type IV: Multidimensional numerical anomaly</p> <p>a) Peripheral point b) Enclosed point c) Local density anomaly d) Global density anomaly e) Local additive anomaly f) Deviant numerical spatial point (typically in images) g) Deviant numerical spatio-temporal point (typically in videos)</p> <p>Type V: Multidimensional categorical anomaly</p> <p>a) Uncommon class combination b) Deviant categorical vertex c) Deviant categorical edge</p> <p>Type VI: Multidimensional mixed data anomaly</p> <p>a) Incongruous common class b) Incongruous common sequential class c) Deviant vertex d-f) Unusual vertex insertion/change/removal g) Deviant edge h-j) Unusual edge insertion/change/removal k) Deviant spatial point (typically in geo data) l) Deviant spatio-temporal point (typically in geo data)</p>
	Multivariate	<p>Quantitative attributes</p> <p>Type VII: Aggregate numerical anomaly</p> <p>a) Deviant cycle b) Temporary change c) Level shift d) Innovational outlier e) Trend change f) Variation change g) Deviant numerical spatial region (typically in images) h) Deviant numerical spatio-temporal region (typically in videos) i) Point-based aggregate anomaly j) Distribution-based aggregate anomaly</p> <p>Type VIII: Aggregate categorical anomaly</p> <p>a) Deviant class aggregate (typically in texts) b) Deviant categorical subgraph c) Deviant relational aggregate</p> <p style="text-align: center;"><i>Atomic multivariate anomaly</i></p> <p>Qualitative attributes</p> <p>Type VIII: Aggregate categorical anomaly</p> <p>a) Deviant class aggregate (typically in texts) b) Deviant categorical subgraph c) Deviant relational aggregate</p> <p>Mixed attributes</p> <p>Type IX: Aggregate mixed data anomaly</p> <p>a) Class change b) Deviant class cycle c) Deviant class sequence d-j) Deviant spatio-temporal/variation sequence amplitude/trend/variation sequence k) Deviant subgraph l) Deviant subgraph m) Appearing/disappearing/flickering/merging/splitting/growing/shrinking/eccentric (sub)graph n) Deviant spatial region (typically in geo data) o) Deviant spatio-temporal region (typically in geo data) p) Point-based mixed data aggregate anomaly q) Distribution-based mixed data aggregate anomaly</p> <p style="text-align: center;"><i>Aggregate anomaly</i></p>	<p>Quantitative attributes</p> <p>Type VII: Aggregate numerical anomaly</p> <p>a) Deviant cycle b) Temporary change c) Level shift d) Innovational outlier e) Trend change f) Variation change g) Deviant numerical spatial region (typically in images) h) Deviant numerical spatio-temporal region (typically in videos) i) Point-based aggregate anomaly j) Distribution-based aggregate anomaly</p> <p>Type VIII: Aggregate categorical anomaly</p> <p>a) Deviant class aggregate (typically in texts) b) Deviant categorical subgraph c) Deviant relational aggregate</p> <p>Mixed attributes</p> <p>Type IX: Aggregate mixed data anomaly</p> <p>a) Class change b) Deviant class cycle c) Deviant class sequence d-j) Deviant spatio-temporal/variation sequence amplitude/trend/variation sequence k) Deviant subgraph l) Deviant subgraph m) Appearing/disappearing/flickering/merging/splitting/growing/shrinking/eccentric (sub)graph n) Deviant spatial region (typically in geo data) o) Deviant spatio-temporal region (typically in geo data) p) Point-based mixed data aggregate anomaly q) Distribution-based mixed data aggregate anomaly</p>

FIGURE 1.2: Anomaly type categorization by Foorthuis et al. [19, p.11]

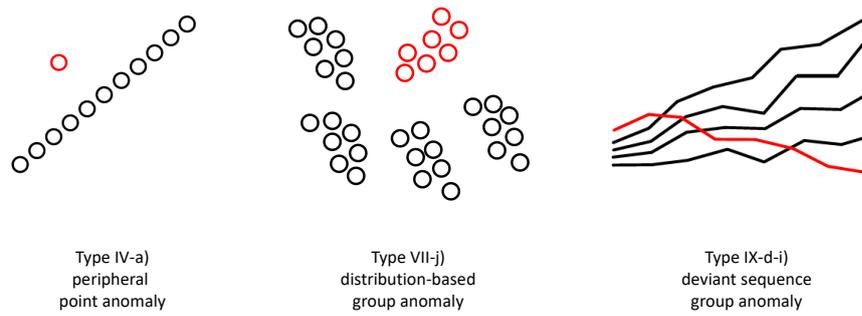


FIGURE 1.3: Thesis relevant anomaly subtypes by Foorthuis et al. [19, p.11]

Chapter 2

Research Contributions

Chapter 2.1 provides an overview of the structure of this thesis as well as the contributions achieved regarding the related papers.

Chapter 1.2 (Anomaly Detection) provided the background knowledge for the algorithms developed for Point Anomaly Detection (Chapter 2.2) and for Group Anomaly Detection (Chapter 2.3), whereas Chapter 2.4 (Evaluation) describes the background for the OAB Framework. Chapter 2.5 (Outlier robustness and applications) describes the scope for the related thesis paper. Chapters 3 - 8 contain the contributed thesis papers.

2.1 Research Questions, Contributions and Thesis Structure

In this thesis, six **research questions** are answered by related thesis papers.

Point Anomaly Detection (cf. Section 2.2) covers among others PCA-based methods, e.g. [43], which score the abnormality of points by the distance to their principle components. During PCA, they calculate the covariance for the covariance matrix, which uses the outlier sensitive mean leading to less distribution aligned eigenvectors and with that to less accurate anomaly scores. An outlier robust³ alternative to the outlier sensitive mean is the median, which can be utilized for PCA as part of the so-called CoMAD matrix (instead of covariance matrix) by calculating the co-median (CoMAD) instead of the covariance. Hence, we raise the following research question **RQ1** to explore the robustness³ of CoMAD for an anomaly detection algorithm.

Since anomaly detection algorithms as so-called batch algorithms do not allow requests of intermediate results, we also raise the following research question **RQ2** to investigate, if anomaly detection algorithms can also provide results at any time.

RQ1 How can CoMAD be used to robustify anomaly detection towards outliers?

RQ2 How can anomaly detection algorithms respond at any time?

Group Anomaly Detection (cf. Section 2.3) covers besides others deep learning based methods, which use neural networks for group feature extraction and outlier scoring in an end-to-end manner. Although there are existing deep GAD approaches, e.g. based on VAEs [10], there is, best to our knowledge, no transformer based architecture as approach for GAD in common and with that neither for GAD on trajectories. Hence, we raised the following research question **RQ3** to explore transformer architectures and also their transparency for GAD on trajectories.

Similar to trajectories, bike rental returns provide reasonable pattern for GAD as so-called distribution-based group anomalies (cf. Section 1.1). These can be modeled as so-called mean embeddings [37], but since these are location dependent, we investigated in a location-free encoding for GAD on bike rental returns. Hence, we raise research question **RQ4**.

RQ3 How can abnormal trajectories be detected as group anomalies?

RQ4 How can abnormal bike rentals be detected as group anomalies?

Anomaly Detection Evaluation (cf. Section 2.4) is conducted by various state of the art benchmarks like [16, 8, 15], but none of them offers a framework for standardized reproducible benchmarking for anomaly detection algorithms on tabular and image data. Hence, we raise the following research question **RQ5** to investigate how the efforts for developing anomaly detection algorithms can be reduced.

RQ5 How can anomaly detection algorithm benchmarking be standardized and made reproducible?

Hyper-Spectral Image (HSI) models (cf. Section 2.5) utilize e.g. 2D convolutions [20, 9] to extract pattern from the wavelengths of hsi images, but there exists, best to our knowledge, no 3D convolutional approach for soil parameter estimation on hsi image data capable to robustly³ cope with outlier wavelengths. Hence, we raise the following research question **RQ6**.

RQ6 How can regression models predict outlier robustly soil parameters from satellite based hyperspectral image data?

The **contributions** of this thesis can be summarized as follows:

- Novel point anomaly detection algorithm variants (CoMadOut) based on outlier robust coMAD PCA (**RQ1**) are proposed as unsupervised algorithms in the work of Andreas Lohrer, Daniyal Kazempour, Maximilian Hünemörder, and Peer Kröger. “CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD”. in: *Machine Learning* (May 2024). ISSN: 1573-0565. DOI: 10.1007/s10994-024-06521-2. URL: <https://doi.org/10.1007/s10994-024-06521-2> - published in *Springer Machine Learning Journal (MLJ)* 2024
- An Anytime-algorithm (AnyCORE) for cluster-based point anomaly detection and removal (**RQ2**) is proposed as unsupervised algorithm in the work of Andreas Lohrer, Anna Beer, Maximilian Archimedes Xaver Hünemörder, Jenny Lauterbach, Thomas Seidl, and Peer Kröger. “AnyCORE - An Anytime Algorithm for Cluster Outlier REmoval”. In: *Proceedings of the LWDA 2021 Workshops: FGWM, KDML, FGWI-BIA, and FGIR, Online, September 1-3, 2021*. Ed. by Thomas Seidl, Michael Fromm, and Sandra Obermeier. Vol. 2993. CEUR Workshop Proceedings. CEUR-WS.org, 2021, pp. 145–156. URL: <https://ceur-ws.org/Vol-2993/paper-14.pdf>
- A novel transparent transformer encoder architecture (GADFormer) for group anomaly detection on trajectories (**RQ3**) is proposed as un- and semisupervised algorithm in the work of Andreas Lohrer, Darpan Malik, Claudius Zelenka, and Peer Kröger. “GADformer: A Transparent Transformer Model for Group Anomaly Detection on Trajectories”. In: (2024). arXiv: 2303.09841 [cs.LG] - accepted at *IEEE International Joint Conference on Neural Networks (IJCNN) 2024, Yokohama, Japan*
- A novel covariance vector encoding for group anomaly detection on bicycle rental movements (**RQ4**) is proposed as unsupervised algorithm in the work of Andreas Lohrer, Johannes Josef Binder, and Peer Kröger. “Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities”. In: *Proceedings of the 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science, IWCTS 2022, Seattle, Washington, 1 November 2022*. Ed. by Andy Berres, Kuldeep R. Kurte, and Haowen Xu. ACM, 2022, 14:1–14:4. DOI: 10.1145/3557991.3567801. URL: <https://doi.org/10.1145/3557991.3567801>
- An open anomaly benchmark framework (OAB) for testing anomaly detection algorithms in un- and semisupervised settings (**RQ5**) is proposed in the work of Andreas Lohrer, Jan Deller, Maximilian Hünemörder, and Peer Kröger. “OAB - An Open Anomaly Benchmark Framework for Unsupervised and Semisupervised Anomaly Detection on Image and Tabular Data Sets”. In: *2021 International Conference on Data Mining, ICDM 2021 - Workshops, Auckland, New Zealand, December 7-10, 2021*. IEEE, 2021, pp. 991–1000. DOI: 10.1109/ICDMW53433.2021.00129

- A 3D convolutional architecture (SpectralNet3D) for soil parameter prediction based on outlier robust preprocessing and Huber-Loss (**RQ6**) is proposed as supervised algorithm in the work of Claudius Zelenka, Andreas Lohrer, Mirjam Raffaella Bayer, and Peer Kröger. “AI4EO Hyperview: A SpectralNet3D and RNNPlus Approach for Sustainable Soil Parameter Estimation on Hyperspectral Image Data”. In: *2022 IEEE International Conference on Image Processing, ICIP 2022, Bordeaux, France, 16-19 October 2022*. IEEE, 2022, pp. 4263–4267. DOI: 10.1109/ICIP46576.2022.9897889. URL: <https://doi.org/10.1109/ICIP46576.2022.9897889>

A visual illustration for the thesis structure is provided in Figure 2.1. The papers contributed by this thesis are highlighted within boxes with black background.

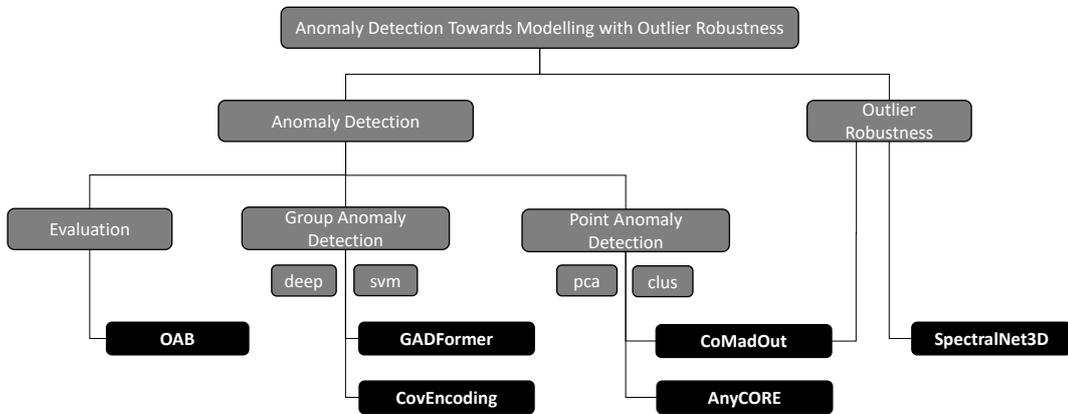


FIGURE 2.1: Structural Overview

2.2 Point Anomaly Detection

The following chapters locate the contributed works in relation to the PAD variants described in Chapter 1.2.

2.2.1 CoMAD PCA-based Point Anomaly Detection

Peripheral Outliers, as described as Type IV-a) in chapter 1.1, are characterized by their high distance from the location of normal points. These distant outliers have a potential negative influence on the predictions of a machine learning algorithm. They especially limit PCA [25]-based methods with outlier sensitive mean by distorting the rotation and translation of the principal components. As a consequence, principal components match less accurately with the underlying distributions in scale and orientation, leading to wrong classification of outliers and non-outliers.

Since these limits make it more challenging for the machine learning algorithm to detect such peripheral outliers, it is highly relevant to investigate in a solution for this problem.

Hence, in order to tackle the described problem, an algorithm being robust³ towards outliers but capable of detecting outliers would be desirable and a suitable solution.

Addressing research question **RQ1**, we investigated in the related thesis paper in CoMAD PCA [41, 26] how to obtain the desired robustness towards outliers and experimented with various scoring methods to fine-tune the algorithm's capability to detect outliers. Both coMAD PCA and scoring methods allow to find an appropriate boundary focused alignment (cf. Fig.5 in paper [33]) of the degree of outlierness between normal and abnormal instances, so that our thesis contribution CoMadOut [33] could achieve an on par performance compared to other unsupervised point anomaly detection algorithms like Isolation Forest [30], LOF [7], etc.

For more details, please see the contributed thesis paper in chapter 3.

Challenges. There were also challenges during the design and development of CoMadOut. One of them was to identify suitable measures of in- and out-of-distribution to achieve a distribution boundary focused outlier scoring. Moreover, large and highdimensional datasets led to runtime- and memory-intensive computations, for which a solution had to be found.

2.2.2 Cluster-based Anytime Point Anomaly Detection

Point anomalies do not only appear far from normal cluster centroids, but can also be located as noise and outliers between clusters. Outliers and so-called in-between-instances cannot clearly be assigned to specific clusters and can thereby cause an additional negative impact on the results of a clustering algorithm, e.g. in terms of silhouette coefficient. Anomaly detection algorithms allow to detect these outliers for a potential removal, but they usually work in batch mode and do not provide their results before they are completely finished. As a consequence, this would result in long waiting times and would thus block in-time user reactions since today's application domains often accommodate large and high-dimensional amounts of data leading to slow response times.

A solution for these limitations would enable an algorithm usage in real-time applications allowing domain experts to immediately analyze the results and react in time. Furthermore, these intermediate cluster results can profit (e.g. in terms of silhouette coefficient) from upfront detected and subsequently removed anomalies.

In order to obtain not only a best possible clustering result, but also to obtain it at anytime, an ensemble of clustering algorithm and anomaly detection algorithm would be desirable, which clusters and optimizes the data iteratively to provide improved results at anytime.

Therefore, we researched with AnyCORE [32] (Anytime Cluster Outlier Removal) an anytime version of the unsupervised point anomaly detection algorithm MORE++ [4] addressing research question **RQ2**.

³In this work the term robustness means insensitivity towards negative impacts of outliers for the final algorithm output.

For more details, please see the contributed paper in chapter 4.

Challenges. During the design and development of AnyCORE we had to tackle several challenges. One of them was to identify a suitable clustering algorithm which benefits from iterative outlier removal. Furthermore, large and highdimensional datasets led to runtime-intensive computations.

2.3 Group Anomaly Detection

The following chapters introduce two of the most popular categories of GAD methods, describe their main properties and locate the contributed papers in relation to them.

2.3.1 Deep Group Anomaly Detection

Deep Group Anomaly Detection approaches utilize deep architectures like Variational Autoencoder (VAE), Adversarial Autoencoder (AAE) [10] or Graph Autoencoder (GAE) [12] for group-based feature extraction. Therefore, feature vectors are usually composed of predefined group structures (e.g. $\mathcal{G} = \{o_1, o_2, \dots, o_n\}$) acting as input for these architectures. Since group members can be arbitrary data entities, a usage for images, texts, videos or others is possible for GAD methods. Moreover, sequences like trajectories, whose trajectory steps have a group relation to their whole trajectory, are a suitable candidate for GAD approaches and can be detected as deviant sequence group anomalies as described as Type IX-d-i) in chapter 1.1 and Figure 1.3.

Reviewing the corresponding literature revealed that deep neural architectures are not yet sufficiently investigated for group anomaly detection. Towards closing this research gap, we recognized similarly to [22] that deep neural architectures are a transparency lacking black box. This fact requires our attention while finding a solution because achieving at least partial model transparency (e.g. by visualizing the capability of model blocks to distinguish between outliers and non-outliers) would allow to support the search for suitable hyperparameters and model architectures.

Furthermore, well-established deep sequence models have difficulties learning long-term dependencies within sequences [24], requesting a further approach to overcome this drawback. In addition to that, sequence data like trajectories are highly available to train deep architectures but often lack ground truth information [46], so that suitable anomaly detection algorithms should opt for an unsupervised or semi-supervised learning setting.

To further tackle the aforementioned problems, an anomaly detection algorithm should be able to learn contexts within groups and, as a result, be able to learn intra-group dependencies. Moreover, it should provide artifacts like e.g. an attention matrix, which allows to improve the model transparency.

With increasing success of transformer models for word sequences [29] a BERT [14]-based transformer encoder approach also proved to be a suitable choice for GAD on trajectories which we demonstrated by our un- and semisupervised deep GADFormer [34] approach and thereby we answered research question **RQ3**.

For more details, please see the contributed paper in chapter 5.

Challenges. During the design and development of GADFormer several challenges had to be tackled. First of all, optimization potential of each model block during architecture search as well as a suitable loss objective had to be identified. In addition to that, a solution in case of different trajectory lengths was necessary. Furthermore, in order to achieve model transparency, a meaningful and robust representation for outlier attention matrices had to be defined.

2.3.2 OCSVM-based Group Anomaly Detection

Originated in the application domain of the bike rental service of the Munich Transport Association (MVG)⁴, whose bike rental service meets with little response in the population [38], we investigated for further methodologies to detect distribution-based bike rental group anomalies. These distribution-based group anomalies are described as Type VIIj) in chapter 1.1. Figure 1.3 depicts the abnormal group of this type with a different covariance in red, whereas the normal groups are black.

Current GAD methods utilize so-called mean embeddings [37] to detect distribution based group anomalies but the patterns of the MVG problem domain are not dependent on a specific location. Furthermore, the usage of point anomaly detection methods as described in Chapter 2.2 would not be capable of detecting group anomalies without specific distribution-based group feature extraction.

From a domain point of view, a solution to this problem would not only have economical benefits, but would also contribute to sustainability due to an increased bike usage among the population by an improved urban planning. From an algorithm point of view, one would also be able to detect distribution-based group anomalies, which are not mapped to the distribution mean.

In order to solve the described problems and to answer research question **RQ4**, we contributed a location independent covariance-based group feature vector representation

$$x' = (\sigma_{11}, \sigma_{12}, \sigma_{21}, \sigma_{22}) \quad (2.1)$$

as input for the unsupervised anomaly detection algorithm OCSVM [42]. This allowed us to detect regions in Munich with abnormal bike rental behavior, thereby providing suggestions for improving bike accessibility, which optimizes the usage of the MVG service. Additionally, we introduced the MVG dataset as a public dataset for group anomaly detection.

⁴<https://www.mvg.de/services/mvg-rad.html> (online: 16.02.2024)

For more details, please see contributed paper [31] in chapter 6.

Challenges. We faced challenges during the design and development of this paper approach. One of them was to identify suitable statistical property for distribution-based group anomaly. Moreover, small group sizes led to less meaningful results.

2.4 Evaluation

The process steps of the design and development of machine learning algorithms are usually quite close to these of the KDD process [17] which are

1. Data Selection
2. Preprocessing
3. Transformation
4. Data Mining
5. Evaluation

In the data selection phase, the researcher defines the datasets for which the machine learning algorithm should proof a sufficient performance. This step can be guided either by the application domain, which the algorithm is for, or by public standard datasets, to which related approaches compared themselves to, due to appropriate performance comparability. In the subsequent preprocessing phase, the data can be handled regarding duplicates or missing values, whereas in the next phase possible transformations like one-hot encoding for categorical values, normalization or standardization are conducted. The data of these previous steps are the baseline for the sampling phase in which heterogeneous strategies can lead to different algorithm performances. In this phase, factors like anomaly contamination rate, up-/downsampling, train/test split rate, number and choice of seeds can have a beneficial but also a negative effect on an algorithms performance. Furthermore, the choice of model- and hyperparameters and finally the choice of reasonable performance metrics are relevant factors for a meaningful algorithm evaluation.

These numerous steps lead to challenging circumstances for reproducibility and comparability. Moreover, these steps imply recurring efforts for each new algorithm development (e.g. data selection and preprocessing, hyperparameter definition, etc.), keeping the researcher off the actual algorithm design.

Solving these problems and reducing these efforts would accelerate the research for machine learning algorithms and shorten the time until appropriate data-driven solutions are ready to use for various application domains.

In order to fill this gap of standardization and to motivate researchers to contribute to the anomaly detection research domain, we contributed with

the thesis paper OAB [35] an Open Anomaly Benchmark Framework for unsupervised and semi-supervised anomaly detection for tabular and image data. Thereby we support the goals of the reproducibility check list for publications [39] published by Pineau et al. and also answered research question **RQ5**.

For more details, please see the contributed paper in chapter 7.

Challenges. During the design and development of OAB, several challenges had to be tackled. One of them was to identify all criteria necessary for reproducibility as well as to define a scope for supported algorithms and datasets. Furthermore, a solution had to be found to allow low effort extensions to new algorithms and datasets. In addition to that, suitable sampling strategies had to be identified.

2.5 Outlier Robustness and Applications

As described in Chapter 3, noise and outliers can have a positive but also negative influence on machine learning algorithms and can therefore have a beneficial or adverse impact on the related application domains at the same time. Hence, outlier robustness³ for machine learning algorithms is a desirable property we aim for, with our contributed thesis approach CoMad-Out [33].

In the agricultural application domain, in which soil parameters get extracted with the help of hyperspectral image data from satellites, noise and outliers are involved and ground truth information first needs to be collected manually from the agricultural areas, and afterwards has to be cumbersome evaluated by researchers in laboratories. Hence, it can be recognized that the process of soil parameter extraction for a targeted fertilization is rather costly and implies huge efforts without exploiting the potential of applying a machine learning algorithm. An overview of this application domain can be seen in Figure 2.2.

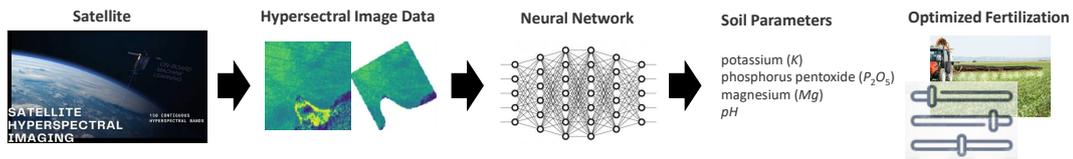


FIGURE 2.2: Application domain overview of SpectralNet3D.

The hyperspectral image data has the form of a cube and consists of 150 masked bands, whereas each image has different band image sizes and can contain noisy or outlier signals compared to its neighboring bands. For an illustration of the original hyperspectral image data, please see Figure 2.3 on the left side.

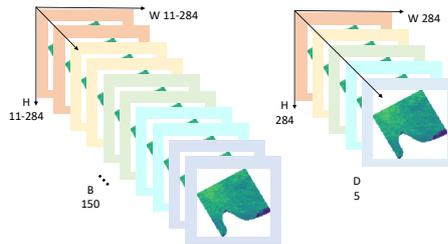


FIGURE 2.3: Left: original bandwidth, right: averaged bands.

The noise and outliers in the band images can be reduced by averaged band ranges as depicted in Figure 2.3 on the right, which also reduces the computational costs for a machine learning model.

Furthermore, the hsi image data model would profit from a loss function being robust to outliers but simultaneously sensitive to small residuals. Although the hsi image data is 3D shaped, there exists no 3D convolutional approach for mapping the features of the bands onto continuous valued soil parameters. Hence, we introduced with our contributed thesis paper [50] not

only a RNNPlus approach for this supervised regression task, but with SpectralNet3D we also introduced a neural network Φ capable of performing the following feature mapping by 3D convolutions:

$$\Phi : X \in \mathbb{R}^{C \times D \times H \times W} \mapsto \mathbb{R}^4 \quad (2.2)$$

and thereby addressed research question **RQ6**.

For more details, please see the contributed paper in chapter 8.

Challenges. There were also challenges during the design and development of SpectralNet3D. First of all, outliers in the single bands had to be smoothed, for which a robust representation had to be identified. Moreover, different image sizes had to be handled without performance loss. Furthermore, a suitable loss objective capable of addressing small residuals and outlying samples simultaneously had to be found. In addition to that, the differences between the distributions between validation and test set turned out to be challenging.

Chapter 3

CoMadOut - A Robust Outlier Detection Algorithm based on coMAD

This chapter consists of a preprint version of the following publication:

Andreas Lohrer, Daniyal Kazempour, Maximilian Hünemörder, and Peer Kröger. “CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD”. in: *Machine Learning* (May 2024). ISSN: 1573-0565. DOI: 10.1007/s10994-024-06521-2. URL: <https://doi.org/10.1007/s10994-024-06521-2>

- **Conception:** Lohrer (Lead), Kazempour (Support), Kröger (Support)
- **Planning:** Lohrer (Lead), Kazempour (Support)
- **Execution:** Lohrer (Lead)
- **Manuscript:** Lohrer (Lead), Kazempour (Support), Hünemörder (Support), Kröger (Support)

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD

Andreas Lohrer^{1*}, Daniyal Kazempour¹, Maximilian
Hünemörder¹ and Peer Kröger¹

¹Christian-Albrechts-Universität zu Kiel,
Christian-Albrechts-Platz 4, Kiel, 24118, Schleswig-Holstein,
Germany.

*Corresponding author(s). E-mail(s): alo@informatik.uni-kiel.de;
Contributing authors: dka@informatik.uni-kiel.de;
mah@informatik.uni-kiel.de; pkro@informatik.uni-kiel.de;

Abstract

Unsupervised learning methods are well established in the area of anomaly detection and achieve state of the art performances on outlier datasets. Outliers play a significant role, since they bear the potential to distort the predictions of a machine learning algorithm on a given dataset. Especially among PCA-based methods, outliers have an additional destructive potential regarding the result: they may not only distort the orientation and translation of the principal components, they also make it more complicated to detect outliers. To address this problem, we propose the robust outlier detection algorithm CoMadOut, which satisfies two required properties: (1) being robust towards outliers and (2) detecting them. Our CoMadOut outlier detection variants using comedian PCA define, dependent on its variant, an inlier region with a robust noise margin by measures of in-distribution (variant CMO) and optimized scores by measures of out-of-distribution (variants CMO*), e.g. kurtosis-weighting by CMO+k. These measures allow distribution based outlier scoring for each principal component, and thus, an appropriate alignment of the degree of outlierness between normal and abnormal instances. Experiments comparing CoMadOut with traditional, deep and other comparable robust outlier detection methods showed that the performance of the introduced CoMadOut approach is competitive to well established methods related to average precision (AP), area under the precision recall curve (AUPRC) and area under

2 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

the receiver operating characteristic (AUROC) curve. In summary our approach can be seen as a robust alternative for outlier detection tasks.

Keywords: Anomaly Detection, Outlier Detection, coMAD, PCA, Unsupervised Machine Learning, Robust Statistics.

Mathematics Subject Classification: 68T99 , 68W25 , 62H86 , 62H25 , 62G35

1 Introduction

Anomaly Detection, one of the major fields of unsupervised machine learning, is an integral part of many domains uncovering the deviations of their data generating processes and thus supporting domain experts in their daily work by identifying rare patterns. However, real world datasets are often highly imbalanced due to the rare occurrence of outliers. Therefore, for predictive methods, the existence of outliers represents obstacle and opportunity at the same time. Depending on the machine learning method, it can be considered as an obstacle, because they may distort the precision of the predictions. On the other hand, methods can utilize outliers as an opportunity, since they may reveal irregular or abnormal patterns and therefore interesting insights. From the aforementioned two aspects, one can derive two properties that an anomaly detection algorithm needs to satisfy, namely (1) to be resilient towards outlying data instances, while at the same time (2) being capable to detect them. A preliminary step of many outlier detection approaches (cf. Section 2) is dimensionality reduction. This part is often realized by PCA (Jolliffe, 1986), a technique providing the directions of highest variances within the data. This is achieved by calculating the eigenvectors of the covariance matrix (directions) and the corresponding eigenvalues (variances). Such eigenpairs allow a transformation from the original data to principal components in lower dimensional subspace, and therewith the task of dimensionality reduction and others. However, the usage of the mean dependent covariance matrix makes the standard PCA potentially susceptible towards outliers, which has lead to the creation of robust PCA methods, minimizing the influence of outliers. (Candès, Li, Ma, & Wright, 2011) In order to achieve resilience or robustness for PCA-based algorithms, we need to ensure that principal components and corresponding eigenvalues are barely or not at all influenced by instances located out of distribution. In particular this means that robustness is achieved if abnormal instances do not skew the orientation and translation of the eigenvectors and do not lead to an increase or decrease of the eigenvalues.

In this work we introduce CoMadOut, an unsupervised outlier detection method, which shows robustness among other techniques due to optimized scoring and outlier resistant PCA (Sajesh & Srinivasan, 2012; Kazempour, Hünemörder, & Seidl, 2019). While both works demonstrate that co-median PCA is potentially robust towards anomalies, it lacks the ability to detect

abnormal instances based on comedian matrix in its non-positive-semi-definite form. Therefore, with this paper, we propose a competitive outlier detection approach called CoMadOut which is capable to detect, score and predict outliers.

Since there already exists a plethora of outlier detection methods, we investigate with this work in the performance of CoMadOut against a comprehensive selection of state-of-the-art techniques (cf. section 4) covering both traditional and Deep Anomaly Detection methods.

In summary, this work provides the following contributions:

1. We introduce the robust outlier detection algorithm CoMadOut, whose baseline variant (CMO) derives margin-based decision boundaries by utilizing the noise-resistant eigenpairs of comedian PCA.
2. We proposed additional CoMadOut variants (CMO*), which improve outlier scoring by simultaneously considering measures of out-of-distribution (tailedness) in order to enhance comedian PCA based scoring to arbitrary distributions, e.g. by kurtosis weighted scores of CoMadOut variant CMO+k.
3. We detail, how the proposed CMO* variants can be combined to an ensemble approach for outlier detection (CoMadOut variant CMOEns).
4. We conduct extensive experiments comparing and discussing the performance of our methods against competitors and several real-world datasets.

The remaining work is structured as follows. In section 2, we give an overview of related work. Section 3 introduces our CoMadOut outlier detection method and elaborate several variants of this method. The experiments are presented in section 4, while section 5 concludes the paper.

2 Related Work

In this section, we review the related literature, concerning all approaches intersecting with our approach CoMadOut. Thereby we divide those into three categories: (1) well established traditional outlier detection methods, (2) deep outlier detection methods, and (3) robust estimation and PCA-based methods, where the latter are optimized towards outlier robustness most similar to our proposed method CoMadOut.

2.1 Traditional Methods

The Local Outlier Factor (LOF)(Breunig, Kriegel, Ng, & Sander, 2000) is a density-based outlier detection method. As such it determines if an object is an outlier based on the k NN-neighborhood. Samples which exhibit a significantly lower density in their own local neighborhood compared to the density of other samples and their respective neighborhoods are identified as outliers.

The Isolation Forest (IF)(Liu, Ting, & Zhou, 2012) method is a tree ensemble approach to identify anomalies. The decision trees of that ensemble are

4 CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD

initially constructed by randomly selecting a feature and then performing a random split between its minimum and maximum value recursively. IF is based on the assumption that outliers exhibit a lower occurrence in contrast to "normal" samples making them appear close to the root of a tree with fewer splits necessary.

The One Class SVM (OCSVM)([Crammer & Chechik, 2004](#)) approach separates all samples from the origin by maximizing the distance from a separating hyperplane to the origin. Therefore, kernels can be utilized to transform the samples into a high dimensional and thus better separable space. Consequently, a binary function computes which regions in the original data space exhibit a high density and are therewith labelled with "+1" while all other samples (outliers) are labelled with "-1".

2.2 Deep Outlier Detection Methods

Two prominent neural network architectures that are commonly used for deep anomaly detection are Autoencoders (AE) and Variational Autoencoders (VAE)([An & Cho, 2015](#)), whose low-dimensional latent vector spaces compete with those of PCA([Hinton & Salakhutdinov, 2006](#)). The underlying concept for both is fairly similar. An Encoder is trained to embed each training sample, so that it can be projected into a generally lower dimensional latent space and then reconstructed by a Decoder network. The assumption for anomaly detection is now that if a model is trained well on normal data, samples that are abnormal should be hard to reconstruct with the same network and therefore have a high reconstruction error. A threshold is then used to identify these anomalies. Among the different variations of AE, VAEs are special versions of autoencoders. VAEs first encode the input as distributions over the latent space followed by a sampling of samples from the learned distributions. In essence VAEs fit normal distributions on the data and achieve a separation from abnormal samples. Another well established deep anomaly detection algorithm is DeepSVDD (Deep Support Vector Data Description) ([Ruff et al., 2018](#)). Following the unsupervised AD assumption, that the majority of instances in the training set is expected to be normal, their approach learns the weights of a neural network with the goal to reduce the dimensionality of its inputs and maps these embeddings towards a center c in the representation space while simultaneously minimizing an enclosing hypersphere around c by using quadratic loss and weight decay regularization. The abnormality of representations is scored by their distance to c .

2.3 Robust Estimation and PCA-based Methods

Within this work PCA (Principal Component Analysis) of [Jolliffe](#) is denoted as standard PCA. It is a well-known technique to find patterns in high dimensional data by analysing the variances within the data. As stated in previous chapters, its outlier-sensitive mean of the calculated covariance matrix makes it sensitive towards outliers and thus it is not ideal for subsequent tasks like

outlier detection. Thus, a more robust, mean-free version is required to perform reliably on outlier datasets.

Comedian PCA (Sajesh & Srinivasan, 2012; Kazempour et al., 2019) follows the same goal as other outlier resistant robust PCA methods and the core idea behind it is intriguingly simple: Instead of computing the eigendecomposition of the covariance matrix, which is highly susceptible to outliers, the computation is performed on a comedian matrix (Falk, 1997). Analogously to the covariance matrix, the comedian matrix represents the median absolute deviation from the median for each dimension and each respective pair of dimensions. Therefore, the coMAD-PCA considers the components with the highest deviation to the median, while standard PCA captures the deviation to the mean. Since the mean of standard PCA is generally more sensitive to outliers (Kriegel, Kröger, Schubert, & Zimek, 2008), coMAD-PCA should be less sensitive. That has been shown by Sajesh and Srinivasan and Kazempour et al., who illustrated the robustness of coMAD-PCA towards outliers.

Amongst the methods that also consider robustness is the so called Minimum Covariance Determinant (MCD) (Rousseeuw, 1984). The goal of MCD is to find a subset of samples whose covariance matrix has the minimum determinant. Its location is then the subset's average, and its variance is the subset's covariance matrix. Rousseeuw states in addition that this technique can yield suitable results even when 50% of the data are contaminated with outliers.

Since the performance of Empirical Covariance Estimators, like e.g. the performance of the Maximum Likelihood Covariance Estimator (MLE), suffers from distorted eigenvectors when used on datasets with outliers, more robust methods have been developed. They replace the outlier sensitive parts mean and (co)variance by robust alternatives. These alternatives use e.g. the samples of the lowest covariance matrix determinant (MCD) or randomly selected samples (FastMCD) (Rousseeuw & Driessen, 1999)¹ to provide a robust mean and covariance or compute it deterministically (DetMCD) (Hubert, Rousseeuw, & Verdonck, 2010) in order to achieve a robust distance measure and thereby a high breakdown value².

With the goal of becoming robust to outliers, there were several ideas on how to achieve this. At this point the role of coMAD for CoMadOut (cf. section 3 - step 1) shows parallels to the Stahel–Donoho outlyingness (SDO) measure (Maronna & Yohai, 1995), which instead uses a weighted mean vector and covariance matrix, and to the comedian approach (Sajesh & Srinivasan, 2012), which computes robust mahalanobis distances based on a positive-semi-definiteness-corrected comedian matrix and weights them by a χ^2 -distribution-factor to receive a suitable cut-off value instead. Further methods with parallels to the role of coMAD are estimators like LMS (Least Median Squares) (Rousseeuw, 1984) which fits to the minimal median squared distances and the estimator MOMAD (Median of Means Absolute Deviation) (Depersin & Lecué, 2021), which considers the SDO as notion of depth (close to our

¹cf. Elliptic Envelope (<https://scikit-learn.org/stable/modules/generated/sklearn.covariance.EllipticEnvelope.html>)

²proportion of outlier samples an estimator can handle before returning a wrong result

6 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

notion of robust inlierness in section 3.2) while estimating mean values, which are robust to outliers. (Peña & Prieto, 2001) detects outliers by locally optimizing projections according to kurtosis coefficients between a contaminated and an uncontaminated distribution (latter used as robust estimator for mean and covariance) using mahalanobis distance and a MAD-normed distance to the median as measure for outlyingness.

PCA-MAD(Huang, Jin, Yu, & Li, 2021) also address the issue of outlier sensitivity and non-robustness of mean-based PCA by weighting outlier scores based on outlier-sensitive standard PCA projections but MAD-weighted z-score distances instead of outlier resistant and kurtosis-weighted coMAD-PCA projection distances or median based margins as the variants of our approach CoMadOut do (cf. Fig. 4).

With this brief overview on the literature of state-of-the-art outlier detection methods we differentiated existing methods from our approach. Best to our knowledge there are no other approaches like our CoMadOut baseline (cf. section 3 CMO Steps 1-5) using inlier region enhancing coMAD-based orthogonal distance medians as robustness improving noise margins in combination with coMAD-based robust inlier regions to identify, score and predict outliers. Furthermore, there is no approach like our CoMadOut variants CMO* optimizing coMAD-PCA based outlier scores by simultaneously weighting according to variance or tailedness (cf. section 3.6 CMO* Steps 1-3).

3 CoMadOut

With this section we introduce CoMadOut (CMO), an unsupervised robust outlier detection algorithm, which follows the assumption that coMAD-based principal components and a robust measure of in-distribution (ID), median m , as well as measures of out-of-distribution (OOD), kurtosis κ , optimize the alignment of the decision boundary between normal and abnormal instances. Thus we introduce besides the baseline algorithm CMO also its variants CMO*. The following paragraphs provide a step-wise explanation and address similarities and differences between the several CMO variants. An overview is provided by the table in Fig. 1.

The common goal of all CoMadOut variants is to first of all obtain outlier resistant subspace orientations. This is achieved by (1) using coMAD-PCA(Sajesh & Srinivasan, 2012; Kazempour et al., 2019) with their robust comedian matrix(Falk, 1997) instead of standard PCA with its outlier sensitive covariance matrix (cf. Fig. 2). This allows CoMadOut to utilize the outlier resistant eigenvectors and eigenvalues of coMAD-PCA in order to receive a robust subspace representation (a coordinate system with axes spanned by the coMAD-PCA eigenvectors) and with that the initial region for inliers for the baseline approach CMO. Since noise is often modeled as a weak form of outliers (Aggarwal, 2016), CoMadOut improves its outlier robustness and its false positive rate also by (2) an additional noise margin (NM) based on the

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD 7

median of the euclidean orthogonal distances between the samples and their subspace axes k extending the initially calculated inlier region.

In particular CoMadOut can be described by the following steps and variants. All variants have step 1 and 2 in common and get variant-specific starting from step 3, whereas the width of a cell indicates for which algorithm variant in the column header its content is relevant.

Step	Sect.	CoMadOut-Baseline (CMO)	CMO+	CMO+k	CMO+e	CMO+ke	CMOEns	Sect.
1.	3.1	CoMAD-PCA						3.1
2.	3.2	Orthogonal projections on CoMAD-Eigenvectors						3.2
3.	3.3	coMAD-Eigenvector-Inlier-Margins	Outlier Scoring by Out-Of-Distribution measures					3.6
4.	3.4	coMAD-Eigenvector-Noise-Margins	-			obtain max. scores		
5.	3.5	CoMadOut Outlier Detection	-			z-score threshold		

Fig. 1: Overview of CoMadOut (CMO*) variants in bold and their steps.

3.1 Step 1 - Computation of CoMAD-PCA Matrix

Let X be a centered matrix with n samples $x \in \{x_1, x_2, \dots, x_n\}$ having d -dimensional feature vectors in $\mathbb{R}^{n \times d}$ and let the comedian³ matrix $COM(X)$ of (Falk, 1997) be defined by

$$X = \begin{pmatrix} x_{1,1} & \dots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \dots & x_{n,d} \end{pmatrix}, \quad COM(X) = \begin{pmatrix} com(A_1, A_1) & \dots & com(A_1, A_d) \\ \vdots & \ddots & \vdots \\ com(A_d, A_1) & \dots & com(A_d, A_d) \end{pmatrix} \quad (1)$$

with A_k being the k -th feature of samples $x_{*,k}$ and with co-median

$$com(A_i, A_j) = med((A_i - med(A_i))(A_j - med(A_j))) \quad (2)$$

so that the comedian matrix $COM(X)$ acts as counterpart for the covariance matrix Σ in the original PCA, whereas

$$A_i - med(A_i) \quad (3)$$

represents the robustness providing subtraction of the median in place of the subtraction of the outlier sensitive mean. On the resulting comedian matrix we apply PCA like (Sajesh & Srinivasan, 2012; Kazempour et al., 2019) in order to utilize its resulting robust eigenpairs to define the selective inlier region for the CMO baseline. Therefore, we consider

$$\delta(X) = U\Lambda U^T \quad (4)$$

³due to the ambiguous usage of COM for co-median and comedian matrix the term coMAD matrix from (Kazempour et al., 2019) has been used interchangeably

8 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

with eigenvector matrix U and eigenvalue matrix $\Lambda (=COM(X))$, where those eigenpairs (consisting out of eigenvector and eigenvalue) with the k -largest eigenvalues are used for the definition of the initial inlier region (for CMO baseline) in the next step 2. Due to the known non-positive-semi-definiteness of the comedian matrix a part of the score-relevant eigenvalues is negative. The approach of [Sajesh and Srinivasan](#) addressed this issue by replacing the original comedian-based eigenvectors and eigenvalues of δ with iteratively adjusted projections resulting in an approximated but positive-semi-definite surrogate matrix. For our CoMadOut approaches we do not want to lose the properties of the original comedian matrix. Therefore, we keep the original non-positive-semi-definite comedian matrix and assume symmetrical gaussian distributed projections so that the sign of the usually small part of negative eigenvalues can be neglected just for the scope of final outlier scoring. Hence, the magnitude of all eigenvalues can get equally included into outlier scoring by taking their absolute value and adjusting the scoring accordingly.

$$\lambda = |diag(\Lambda)| \quad (5)$$

This allows us to base our outlier scoring still on the original robust eigenvectors by additionally including a small adapted eigenvalue part.

Since the ideal choice of K for the number of principal components can vary between datasets, we conducted the experiments of our work (cf. section 4) on different percentages of the total number of L possible subspaces (leading to a different number of K but maximal L principal components) for all datasets and averaged the achieved average performances.

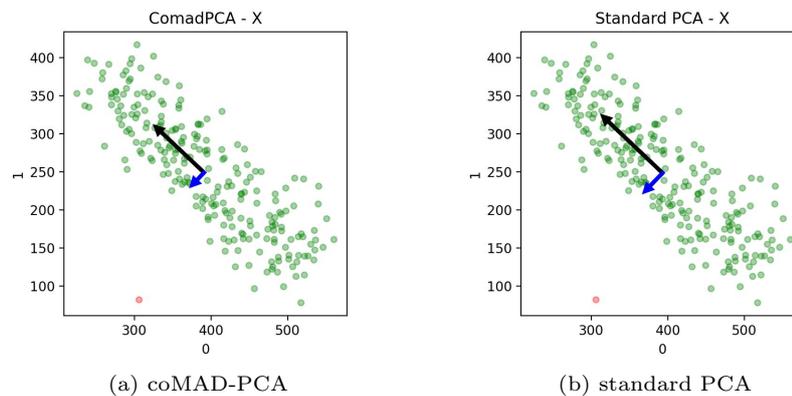


Fig. 2: Illustration of orientation and scale of principal components after applying coMAD-PCA (a) and standard PCA (b) to a synthetic dataset. It can be observed that the outlier (colored in red) has less influence to eigenvectors and eigenvalues of coMAD-PCA on the left than to those of standard PCA on the right.

3.2 Step 2 - Orthogonal projections on coMAD-Eigenvectors

In the scope of CoMadOut the previously calculated eigenpairs are used to calculate the scores of inlierness and outlierness respectively. Therefore, the orthogonal projections of all samples x are calculated for each eigenvector (=principal component). The eigenpairs define the direction (k -th eigenvector \vec{u}_k of matrix U) and scale (k -th eigenvalue λ_k of matrix Λ) and correspond to a subspace axis k .

Formally step 2 can be described by the following equations: $\forall x \in X$ the projections x' can be calculated by

$$x' = ((x^T u_k)/(u_k^T u_k)) \cdot u_k \quad (6)$$

Since the origin of the subspace is zero centered after coMAD-PCA from Step 1, the euclidean distances of the projected samples x' to the origin can be calculated as

$$x'_k = \sqrt{\sum_{k=1}^K (x'_k)^2} \quad (7)$$

After the first two steps, which have CMO baseline approach and its variants CMO* in common, the next steps are either related to CMO or to its CMO* variants.

3.3 CMO: Step 3 - Computation of coMAD-Eigenvector-Inlier-Margins

The next step for the margin-based CMO baseline approach is to span an initial region for inliers. The eigenpairs define the direction (k -th eigenvector \vec{u}_k of matrix U) and scale (k -th eigenvalue λ_k of matrix Λ) of the corresponding eigenvectors, which correspond to the subspace axes. In order to find out which samples x are part of the inlier region, only samples are considered, which got projected on subspace axis k between its origin O and the absolute length of its axis-corresponding eigenvector \vec{u}_k . Since the eigenvector originally is an unit vector the actual eigenvector length u_k - and with that the inlier range for each subspace axis - is defined by the scale of the related eigenvalue λ_k . That λ_k spans the range of our inlier region in positive and negative eigenvector direction with $[-\lambda_k, +\lambda_k]$ on each subspace axis k .

Formally, the decision for each sample x' , if it lays within the inlier range of all subspace axis eigenvectors \vec{u}_k , is defined as follows:

$$inl(x')_k = \begin{cases} 1, & \text{if } x'_k \in [-\lambda_k, +\lambda_k] \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

In order to avoid inliers to be considered as outliers when the inlier region defining eigenvalue $\lambda_k=0$ a minimum inlier region epsilon ϵ of 1e-6 is defined.

10 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

In case a sample x' lays within all inlier ranges the product of all subspace-axis- k -related inlier function results

$$inl(x') = \lceil \frac{\sum_{k=1}^K inl(x')_k}{K} \rceil \quad (9)$$

would be equal to 1 (=inlier), otherwise 0 (=non-inlier, cf. section 3.5 step 5).

3.4 CMO: Step 4 - Computation of coMAD-Eigenvector-Noise-Margins

After defining robust inliers in step 3 also noise samples⁴ are considered for adding them to the set of inliers. Therefore, we extend the robust inlier region ranges, which are corresponding to their subspace axis k

$$[-\lambda_k, +\lambda_k] \quad (10)$$

by the median of its euclidean orthogonal distances representing a robust noise margin bandwidth $m_k = med(x'_k)$ leading to new inlier region ranges

$$[-m_k - \lambda_k, +\lambda_k + m_k]. \quad (11)$$

Thereby, the restrictive λ -thresholds receive an extended margin for each principal component k , capable to also cover noise samples as inliers by still using the outlier robust distances median instead of the outlier sensitive distances mean.

3.5 CMO: Step 5 - CoMadOut Outlier Detection

Having the final decision boundaries between inliers and outliers available (cf. step 4), this step describes how the CMO baseline approach is able to *identify*, *score* and *predict* samples x as outliers.

Since the coMAD-PCA enables CoMadOut to create its eigenpairs according to the robust coMAD matrix, outliers are less likely to distort their orientation and scale, which allows to reliably detect inliers in the first phase. Further restrictively selecting noise samples as inliers increases the set of robust inliers and increases thereby the probability that the remaining samples are actual outliers, which as a consequence enables CoMadOut to implicitly identify outliers.

CoMadOut is also capable to provide sample outlier scores sc_i . For this purpose the absolute euclidean distances x'_k of the projected samples x'_i are first reduced by the inlier margin τ_k aiming that actual outlying non-negative distance residuals of the principal components remain. Based on own empirical

⁴in this work samples are considered as noise samples when they are closer to the related subspace axis than its orthogonal distances median m_k but not part of the set of initial robust inliers defined by the coMAD-eigenpairs (cf. section 3.2 step 2)

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD 11

evaluations the best representation for the final outlier score sc_i for a sample x'_i is the mean of its remaining non-negative distance residuals.

$$sc_i = \text{mean}(\max(0, x'_{i,k} - \tau_k)) \quad (12)$$

CoMadOut can also provide softmax scores. For that the median from all λ -scaled scores sc_i for each score dimension k is calculated to represent an in-distribution score. The non-negative subtraction of that score median from all sample scores allows to uncover residual score dimensions which are more likely to represent an outlier sample so that the subsequent softmax function returns for these dimensions higher outlier scores. The highest softmax score among them represents the softmax score of the related sample x'_i .

$$sc_{i,softmax} = \max_i(\text{softmax}(sc_{i,k}/\sqrt{\lambda_k})) \quad (13)$$

Finally the outlier prediction of the CMO baseline is realized by spanning a hyperrectangle or a k -dimensional rectangular cuboid (for $k=2, 3, \dots, K$ principal components) with the help of the outlier resistant coMAD-eigenpairs according to the related scale (k -th eigenvalue λ_k of matrix Λ) and direction (k -th eigenvector \vec{u}_k of matrix U) respectively, which allows to separate between inliers and outliers by outlier threshold τ with $\tau_k = \lambda_k + m_k$

$$\text{outl}(x')_k = \begin{cases} 1, & \text{if } x'_k \notin [-\tau_k, +\tau_k] \\ 0, & \text{otherwise} \end{cases} \quad (14)$$

In case a sample x' exceeds at least one of the K given subspace axis outlier thresholds τ_k , the outlier decision function

$$\text{outl}(x') = \prod_{k=1}^K \text{outl}(x')_k \quad (15)$$

would be equal to 1 (=outlier) and otherwise 0 (=inlier).

For a geometrical intuition of the scoring and labeling of the CoMadOut baseline approach CMO please see Fig. 3. In Fig. 3 CMO is visualized on a 2-dimensional test dataset. The first principal component (PC) is depicted by the blue and turquoise lines in positive and negative direction and the second PC by the purple and green line accordingly. They define a hyperrectangle (or a k -dimensional rectangular cuboid for $K \geq 3$) for the robust inlier region (green background) and get extended by the noise margin (NM)(gray background), which defines the boundary between inliers (green) and outliers (red).

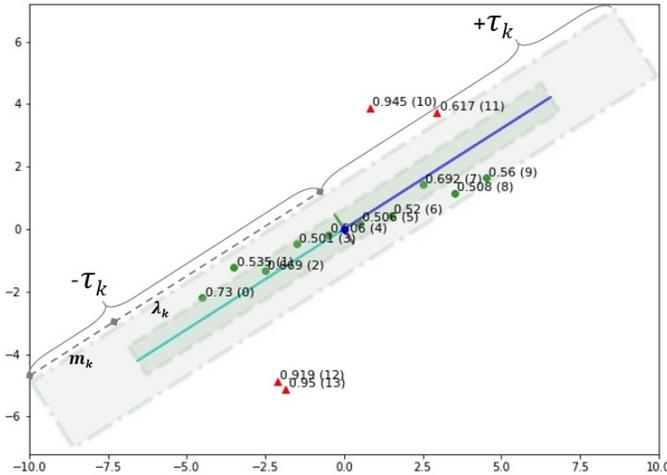


Fig. 3: CoMadOut scoring and labeling of CMO baseline on synthetic data.

3.6 CMO*: Step 3 - Outlier Scoring considering weighting according to out-of-distribution measures

Since the outlier prediction based on the strict inlier and outlier regions of the rectangular cuboid of the CoMadOut baseline approach CMO (cf. Fig. 3) is partially too restrictive for a variety of high-dimensional datasets, further variants of CoMadOut CMO* are introduced with this work. Still utilizing the beneficial properties of coMAD-PCA (cf. sections 3.1 and 3.2 with steps 1 and 2) like outlier resistant eigenpairs or the location of the median as measure of in-distribution, the predictions of variants CMO* are pure outlier scores softening the strict inlier and outlier regions of the CMO baseline.

The related but different approach PCA-MAD of Huang et al. addressed this issue by weighting outlier scores based on outlier-sensitive standard PCA projections (instead of outlier resistant coMAD-PCA projections) and based on principal component based MAD- and median-scaled z-scores.

Our approach CMO+ defines outlier scores based on the sum of absolute distances of coMAD-PCA projections to their zero centered origin O without weighting the outlier scores.

Comparing CMO+ (cf. Fig. 5) and PCA-MAD(Huang et al., 2021) (cf. Fig. 4) one recognizes strong and weak points on both sides. Although the coMAD-based distribution center of CMO+ seems to be well centered and its boundaries are not as geometrically restrictive as those of the CMO baseline it only considers samples quite close to the distribution median as normal. Depending on the domain-specific anomaly semantics of a dataset such outlier scoring could be too restrictive as well. PCA-MAD of Huang et al. has a broader range of normal non-outlier scores around its MAD-scaled mean but struggles like CMO+ to align its outlier scores close to the distribution boundaries of the longitudinally shaped dataset. Scores of border points close to the mean are still representing non-outliers whereas already points in the middle between boundary and mean start to get scored as outliers.

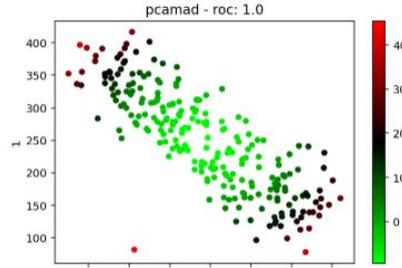


Fig. 4: Outlier score boundaries of competitor PCA-MAD(Huang et al., 2021).

In order to address the issue of neglected distribution boundaries, further CoMadOut variants introduce OOD measures to weight its CMO* outlier scores ($sc_i = x'_i$) accordingly. CMO+k introduces kurtosis based weighting of outlier scores with

$$sc_{i_{CMO+k}} = \kappa \cdot sc_i \quad (16)$$

with kurtosis κ as a measure of dispersion between the two μ and σ (Moors, 1986), and provides with that a notion of distribution tailedness.

Formally kurtosis κ is according to Pearson defined as:

$$\kappa = \mathbb{E} \left[\left(\frac{X - \mu}{\sigma} \right)^4 \right] = \frac{\mu^4}{\sigma^4} \quad (17)$$

where μ represents the distribution mean and σ is the standard deviation.

Considering Fig. 5, the kurtosis based weighting of coMAD-PCA based outlier scores sc_i , introduced with CoMadOut variants CMO+k and CMO+ke, contributes to a distribution boundary focused outlier scoring and thus providing a suitable outlier detection algorithm for related datasets and accordingly demanding domains.

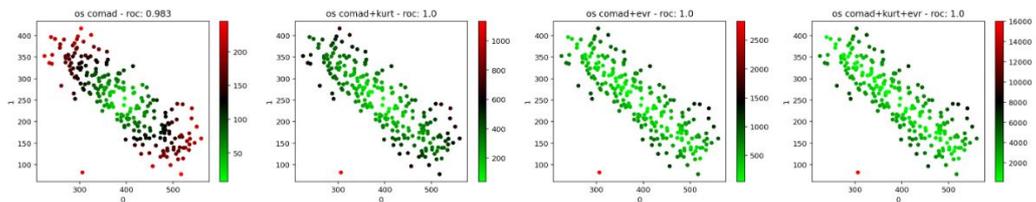


Fig. 5: Outlier score boundaries of CoMadOut variants CMO+, CMO+k, CMO+e, CMO+ke on a synthetic dataset (from left to right).

Since Huang et al. demonstrated with λ -weighted distance outlier scores as part of PCA-MAD a comparable positive effect for boundary adapted outlier scoring, its performance is also investigated with the coMAD-based CoMadOut

14 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

variants CMO+e and CMO+ke whereas the latter includes also kurtosis based weighting.

Formally the outlier scoring of variants CMO* is defined as follows:

$$SC_{i_{CMO+e}} = \frac{SC_i}{\lambda} \quad (18)$$

$$SC_{i_{CMO+ke}} = \frac{\kappa \cdot SC_i}{\lambda} \quad (19)$$

Considering the CMO* scoring in Fig. 5, the applied λ -weighting as introduced in (Shyu, Chen, Sarinnapakorn, & Chang, 2003) with λ as weighting factor, leads to a permanent notion of normality along the direction of the main principal component so that even boundary points in that direction are considered as normal. Comparing the outlier scoring for boundary points, λ -weighted scoring shows due to the principal component related variance less focus to the outlier scores of boundary points in comparison to the pure kurtosis weighted outlier scores.

Investigating also in the combined performance of all CoMadOut variants CMO* this work also introduces its ensemble approach CMOEns which standardizes the outlier scores of CMO* methods by its z-score

$$zSC_{i_*} = \frac{SC_{i_*} - \mu_*}{\sigma_*} \quad (20)$$

and selects for each sample the largest z-score among the outlier z-scores of the CMO* approaches.

$$zSC_{i_{CMOEns}} = \max(zSC_{i_{CMO+}}, zSC_{i_{CMO+k}}, zSC_{i_{CMO+e}}, zSC_{i_{CMO+ke}}) \quad (21)$$

For the ensemble variant CMOEns a sample is considered as an outlier when z-score $zSC_{i_{CMOEns}}$ for a sample exceeds the outlier threshold of $z_{thresh} = 1$:

$$outl(zSC_{i_{CMOEns}}) = \begin{cases} 1, & \text{if } zSC_{i_{CMOEns}} \notin [-z_{thresh}, +z_{thresh}] \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

With this section 3 we introduced the CoMadOut variants CMO and CMO* whose performances are investigated in scope of extensive experiments in the next section 4 comparing them to the performance of related and state of the art methods for unsupervised anomaly detection.

4 Experiments

In order to demonstrate the competitiveness of the CoMadOut variants CMO and CMO*, several experiments have been conducted to measure its capability to detect outliers as well as its runtime behavior.

4.1 Setup

For our experiments we used a paperspace GPU instance with Linux Ubuntu 20.04.3 LTS (Focal Fossa), Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz QuadCore and 45GB RAM memory with usage of 48GB GPUs and parallel processing. Moreover, we wrote our experimental code in Python using Jupyter Notebook.

4.1.1 Datasets

First we want to show how datasets could look like on which our CoMadOut approach works well and on which it starts to fail. For this purpose we created a synthetic toy dataset shown in Fig. 2 with 1 extreme outlier and 235 instances as part of the main distribution of normal instances. We investigated and discussed the algorithm robustness in terms of coMAD-PCA based outlier resistance (cf. details provided with Fig. 4 and Fig. 5 in Section 3.6). In order to further extend the experiments with CoMadOut we also benchmarked the variants of our method on *real world datasets* like the well-known 20newsgroups dataset⁵ representing a large high-dimensional dataset, Boston Housing Prices⁶ dataset, the PARVUS Wine dataset⁷ and many other datasets (cf. Tab. 1) from the ODDS benchmark website⁸ providing more samples and features than the initial synthetic dataset. The Boston Housing Price dataset has been extended by a ground truth label column "outlier" replacing the original regression value target column "MEDV". Therefore and to receive a preferably linear dataset in case of outlier removal, we considered samples laying 2 Inter Quartile Ranges (IQRs) out of the 1st and 3rd quartile as outliers and flagged them with "1" whereas the normal data got flagged with "0". Moreover, the Wine dataset got specifically tailored for the task of outlier detection. In this version from ODDS⁷ samples with label "class 1" (cultivator 1) are reduced to 9 samples and flagged as outliers with label "1". Samples with labels "class 2" or "class 3" are considered as normal samples and flagged as inlier with label "0". Since CMO variants calculate their comedian matrices including all given samples, the 20newsgroup⁵ dataset had to be downsampled to 10% from (n: 11.905, p: 768) to (n: 1.302, p:768) to not exceed the hardware and runtime restrictions but it still represents an high-dimensional real-world dataset.

4.1.2 Compared Methods

With this paper we also want to benchmark our approach against a wide range of state-of-the-art outlier detection algorithms in order to introduce our method as a further competitive and robust alternative to already well established outlier detection algorithms and to emphasize the performance

⁵<http://qwone.com/~jason/20Newsgroups/>

⁶<https://archive.ics.uci.edu/ml/machine-learning-databases/housing/>

⁷<http://odds.cs.stonybrook.edu/wine-dataset/>

⁸<http://odds.cs.stonybrook.edu>

Table 1: Overview of datasets used in our experiments.

dataset	samples	features	outliers
20news	1300	768	65 (5.0%)
arrhythmia	451	274	65 (14.4%)
cardio	1830	21	176 (9.6%)
annthyroid	7199	6	534 (7.4%)
breastw	682	9	239 (35.0%)
letter	1599	32	100 (6.3%)
thyroid	3771	6	93 (2.5%)
mammography	11182	6	260 (2.3%)
pima	767	8	267 (34.8%)
musk	3061	166	96 (3.1%)
optdigits	5215	64	150 (2.9%)
pendigits	6869	16	156 (2.3%)
mnist	7602	100	700 (9.2%)
shuttle	49096	9	3510 (7.1%)
satellite	6434	36	2036 (31.6%)
satimage-2	5802	36	71 (1.2%)
wine	128	13	9 (7.0%)
vowels	1455	12	50 (3.4%)
glass	213	9	9 (4.2%)
wbc	377	30	21 (5.6%)
boston	506	14	179 (35.4%)

of CoMadOut compared to some of its most similar approaches like PCA-MAD(Huang et al., 2021)⁹, Elliptic Envelope¹ and Minimum Covariance Determinant (MCD)(Rousseeuw & Driessen, 1999; Rousseeuw, 1984). For the comparison to non- and semi-robust algorithms we also involved several variants of standard PCA methods (PCA from Shyu et al. and its mean-sensitive covariance matrix in combination with several scoring methods: raw scores - PCA(r), eigenvalue-based scores - PCA(e), scores based on our robust noise margin - PCA(NM), our kurtosis weighted scores - PCA(k) and their combination PCA(ke)). The latter four allow us to directly compare between the impact of comedian and standard PCA for the scoring, once based on robust eigenpairs of CMO(*) approaches and once based on mean-sensitive eigenpairs of PCA(*) approaches. In addition to that also traditional outlier detection methods like LOF(Breunig et al., 2000), KNN(Campos et al., 2016), IsolationForest(IF)(Liu et al., 2012), HBOS(Goldstein & Dengel, 2012) and OCSVM(Crammer & Chechik, 2004), as well as modern Deep Outlier Detection methods like AutoEncoder(AE)(Hinton & Salakhutdinov, 2006), Variational AE(VAE)(An & Cho, 2015) and DeepSVDD(Ruff et al., 2018) had been involved in our

⁹<https://github.com/lohrera/pcamad>

benchmark experiments. As far there existed already reliable implementations of the algorithms in python frameworks like scikit-learn¹⁰, PyOD¹¹ or others⁹ then those had been used. In total, 5 random-states and seed values had been set from 0 to 4 for all experiments, allowing approaches with random effects converge to a stable result. The hyperparameters had been set as follows for the related datasets (cf. Tab. 1). CoMadOut (soft-max_scoring=False, center_by='median'), PCA-based (n_components=0.25, 0.999), AE (hidden_neurons=[4,3,2,2,3,4], batch_size=4, dropout_rate=0.0, epochs=10, l2_regularizer=0.01 for all), VAE (encoder_neurons=[4, 3, 2], decoder_neurons=[2, 3, 4], batch_size=4, epochs=10, dropout_rate=0.0, l2_regularizer=0.001) and DeepSVDD (use_ae=True, hidden_neurons=[64, 32, 4], batch_size=32, epochs=10, dropout_rate=0.0, l2_regularizer=0.1). For all other parameters default values had been used for all experiments. For DeepSVDD we recognized that given default parameters are not sufficient to demonstrate the potential of the approach on tabular data, which could be further improved in future work with dataset specific hyperparameter optimization. So far available, the benchmark results of PCA-MAD(Huang et al., 2021) had been used directly from their paper, otherwise from the non-official implementation⁹. Evaluation metrics described in the subsequent paragraph had been measured with its scikit-learn¹⁰ implementations.

4.1.3 Evaluation Metrics

Outlier detection can be a semi- or unsupervised task conducting binary decisions whether a given sample is normal (=inlier or noise) or abnormal (=outlier). Since the algorithms are mostly threshold-based and the ratio between normal and abnormal data is more likely to be skewed, we use the evaluation metrics Average Precision (AP), Area Under the Receiver Operating Characteristic (AUROC) curve, Area Under Precision Recall Curve (AUPRC) and Precision@n (P@n) with n as the number of total samples.

Especially Precision and Recall allow in presence of highly skewed data a more accurate view on an algorithms performance.(Davis & Goadrich, 2006)

Depending on the relevance for a given domain problem, different evaluation metrics should be considered. In case an anomaly detection method should avoid wrong alerts (false positives) metrics like AUROC should be considered, whereas domains aiming to avoid missed alert (false negatives) should involve AUPRC in their performance analysis. Due to the usually highly imbalanced datasets, metrics like accuracy neglecting an explicit analysis of false predictions are not involved in our evaluation.

4.2 Results

After explaining the experimental setup we introduce our experiment results. The performance of the CoMadOut variants has been compared with that of 26 other outlier detection methods which have been benchmarked with the

¹⁰<https://scikit-learn.org>

¹¹<https://pyod.readthedocs.io>

18 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

datasets and evaluation metrics described in section 4.1. Since the performance of the algorithms based on standard PCA and comedian PCA is dependent on the number of chosen principal components the benchmark can be run on different percentages of the total number of principal components. In order to demonstrate the differences between extreme percentages of e.g. 0.25 (25%) and 1.0 (100%), both benchmark results are reported for each performance metric starting with the results on all principal components (100%).

The metrics in these tables (cf. Appendix A) represent the performance for each combination between algorithm and dataset as well as the average performances ("AVG") of the particular algorithms as well as on a specific dataset. Their row "WIN" indicates how many times an algorithm has achieved the best performance from all benchmarked algorithms. Providing a notion of how much better an algorithm performs compared to all other algorithms the row "ARK" contains Average Rank, which each algorithm achieved on the tested datasets (lower ARK means better). The last row "RK" contains dense absolute ranks starting with 1 as rank for the best algorithm. The cells highlight the best result per algorithm in bold. Results are referenced in brackets related to its paragraph e.g. in paragraph "Results AP & AUROC" the reference (4/8) means (AP-RK100%: 4/AP-RK25%: 8).

Average ranks are the basis for the conducted pair-wise Wilcoxon test with Friedman-Nemenyi post-hoc tests investigating if an algorithm or algorithm group performs significantly better than others or shows no significant performance differences. Inspired by ADBench (Han, Hu, Huang, Jiang, & Zhao, 2022), we calculate a critical difference cd based on the Wilcoxon approach with a significance level of $\alpha = 0.05$, which allows us to illustrate which algorithms perform equally well or better (cf. e.g. the critical difference diagram in the following Fig. 6, where bold line ranges indicate no significant difference between algorithms of this range for the given experimental setting and difference value). The critical difference diagram splits the algorithm list into two halves, the best performing algorithm on the upper left and the comparably worst performing on the upper right. The average rank "ARK" is shown in the diagram right next to an algorithm providing a notion of how much better or worse an algorithm is compared to another one.

4.2.1 Results AP & AUROC

Considering the results related to the evaluation metrics AP (cf. Tab. A1 and Tab. A2) CoMadOut variants perform best ("win") on 5 datasets out of 21 (for 100% and 25% respectively), e.g. CMO+ (2/13) shows on the second best AP rank behind Isolation Forest (1/1) and together with CMO+k (3/6) and CMO+e (3/11) the best performance among all compared PCA-based methods for 100% of all principal components (PCs). Also noticeable is the performance of the standard PCA variants with CoMadOut scoring ("PCA(NM,k,ke,Ens)") on top of covariance matrices achieving wins on 5 and 3 datasets for 100% and 25% PCs. The related critical difference (cd) diagrams in Fig. 6 and Fig. 7

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD 19

confirm these findings, but according to the Nemenyi post-hoc test the performances between the best algorithm IF with ranks (1/1) and e.g. OCSVM with ranks (22/21) at the end of the critical difference range cannot be considered as significantly better ($\alpha > 0.05$). In addition to that, the cd diagrams indicate an insignificant performance loss of many CMO* variants after PC reduction from 100% to 25%, except the kurtosis-weighted variants (CMO+k, CMO+ke), which are comparably stable in terms of less average rank loss and demonstrate with that also its suitability for robust outlier detection after dimensionality reduction.

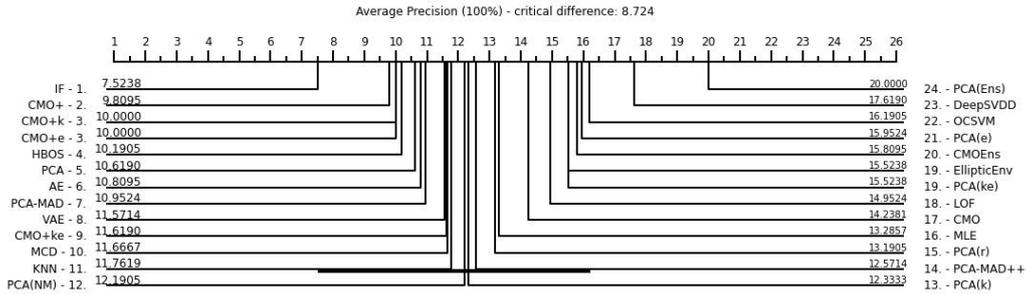


Fig. 6: Average Ranks of Average Precision (AP) performances based on 100% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

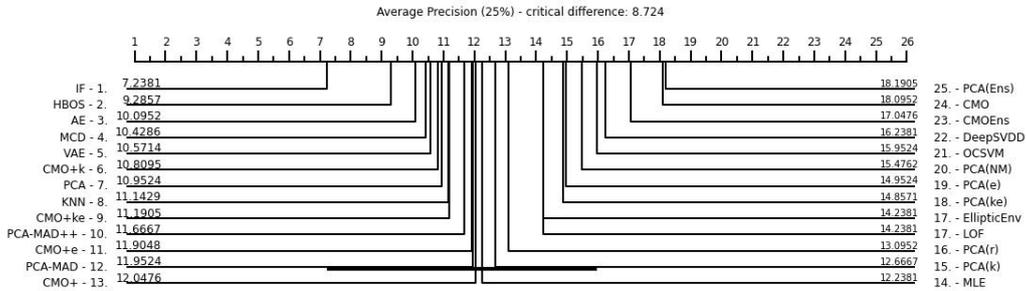


Fig. 7: Average Ranks of Average Precision (AP) performances based on the top 25% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

For AUROC the CoMadOut variants achieve in total 6 best dataset performances on 100% PCs and 5 on 25%. The variants CMO+ (3/12), CMO+k (5/7) and CMO+e (6/9) show the best CMO performances and in case of the first two comparable many total wins as the PCA-MAD (1/2) but are on par with Isolation Forest (4/3) on 100% PCs (cf. Tab. A3). On only the top 25% of all principal components the performance of CoMadOut is comparable with those from the upper midfield (Tab. A4). Considering the cd

20 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

diagrams (cf. Fig. 8 and Fig. 9) PCA-MAD variants outperform all algorithms in each AUROC setting, whereas on 100% PCs the CMO* variants and IF are right behind and on 25% PCs methods as IF, MCD and HBOS. PCA(*) variants also demonstrate by total wins (5/3) that the combination of outlier sensitive covariance matrices and CoMadOut scoring can be beneficial for a selection of datasets, which also includes with 20newsgroup one of the large-scale real-world datasets.

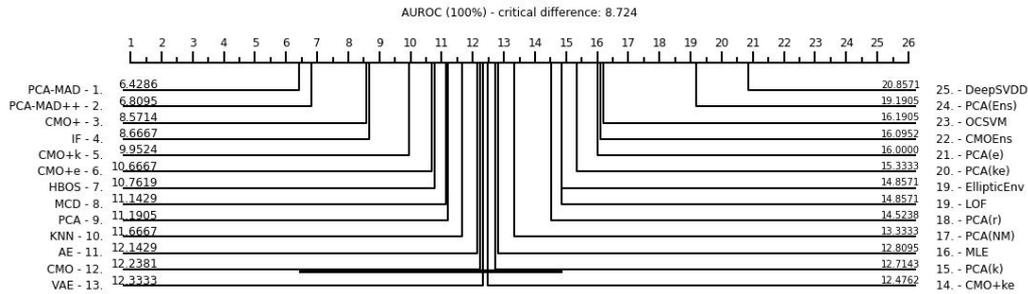


Fig. 8: Average Ranks of AUROC performances based on 100% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

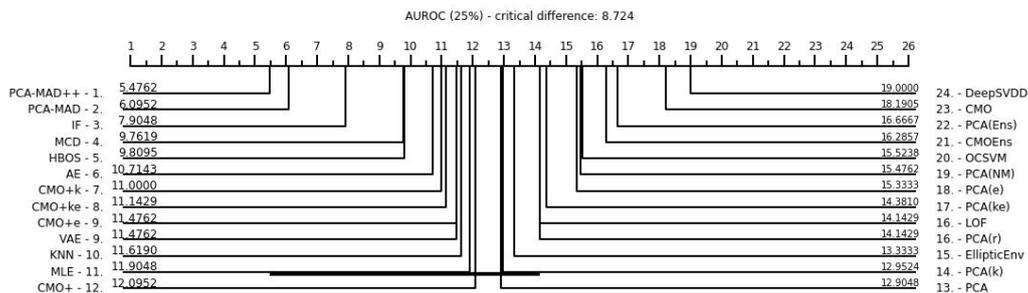


Fig. 9: Average Ranks of AUROC performances based on the top 25% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

4.2.2 Results AUPRC & Precision@n

Comparing the AUPRC performances of the CoMadOut variants, one observes that the CoMadOut variant CMOEns shows the best and second-best performance (1/2) of all algorithms, being on par with Isolation Forest (2/1), and 13 total wins could be achieved by all CMO variants (cf. Tab. A5 and Tab. A6). Furthermore, the scoring variants of CoMadOut could also enhance the scoring for standard PCA methods leading to a total of 7 and 3 wins and PCA(Ens)

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD 21

as third-best (3/3) of all algorithms. Also the performance of CMO+, CMO+k and CMO+e are ranked in the top six on 100% PCs. For comparison of CMO* approaches with all other algorithms the related cd diagrams are shown in Fig. 10 and Fig. 11.

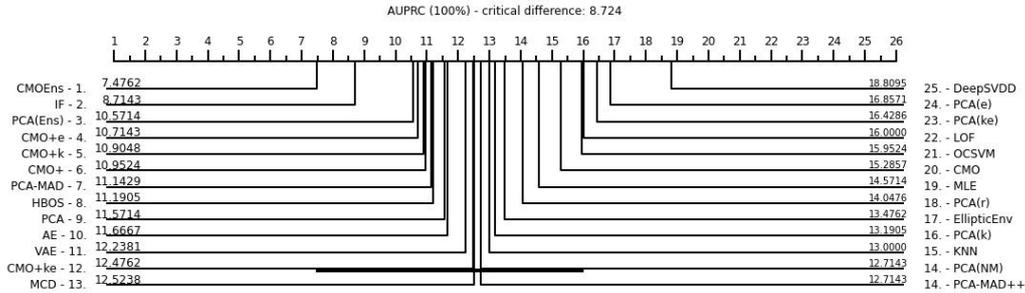


Fig. 10: Average Ranks of AUPRC performances based on 100% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

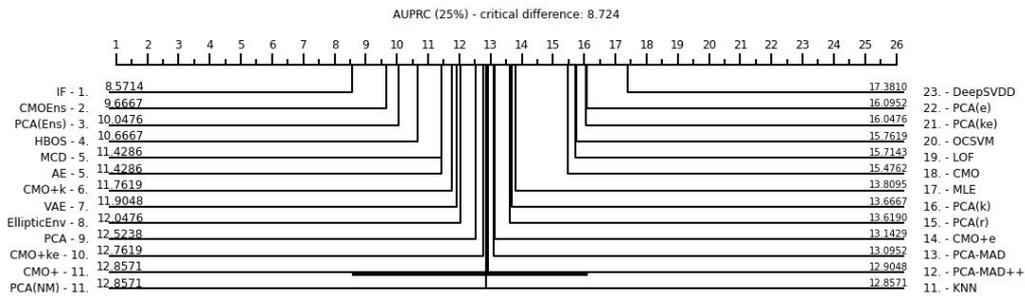


Fig. 11: Average Ranks of AUPRC performances based on the top 25% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

Considering the performances for Precision@n ($P@n$) in Tab. A7 and Tab. A8 the CoMadOut variants achieved a total of 11 best dataset performances on 100% and 25%. Most remarkable is the standard PCA performance with kurtosis weighting of CoMadOut, PCA(k), with a total of 6 best dataset performances. Furthermore interesting are the best dataset performances of methods CMO+k (8/9) and CMO+ (5/10) on 20newsgroup dataset demonstrating its potential for large high-dimensional datasets. The best overall $P@n$ performances achieved PCA-MAD (1/1), PCA-MAD++ (2/2) and Isolation Forest (3/3). Despite these findings the results for the $P@N$ experiments of CMOEns and PCA(Ens) require with zero values further validation. For a detailed comparison of CMO* approaches with all other algorithms the related cd diagrams are shown in Fig. 12 and Fig. 13.

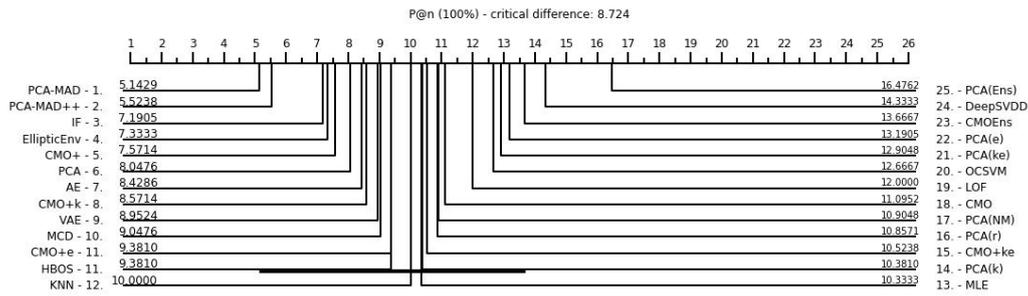
22 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

Fig. 12: Average Ranks of Precision@n ($P@n$) performances based on 100% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

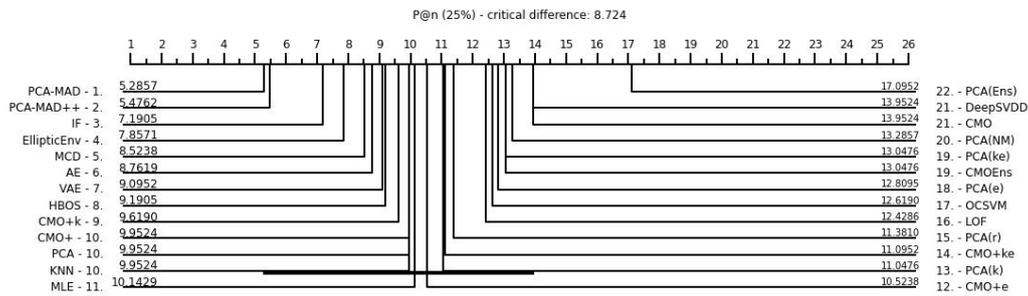


Fig. 13: Average Ranks of Precision@n ($P@n$) performances based on the top 25% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

4.2.3 Runtime

Considering the runtime performances of the CoMadOut variants in Tab. A9 and Tab. A10 one observes that they tend to show rather lower midfield performances despite parallelization. Comparing the runs of the CMO baseline version with newer CMO* variants one recognizes that the choice of variant depends besides its performances on the previously evaluated metrics also on the amount of principal components whether the baseline algorithm or CoMadOuts newer variants are more reasonable. Independent of the amount of principal components the algorithms HBOS (1/2), PCA (2/1) and PCA-MAD++ (3/3) show the fastest performances. For a detailed comparison of CMO* approaches with all other algorithms the related cd diagrams are shown in Fig. 14 and Fig. 15.

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD 23

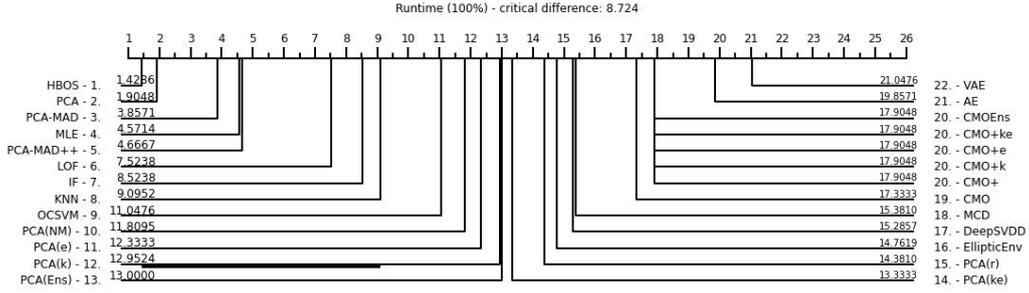


Fig. 14: Average Ranks of Runtime performances based on 100% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

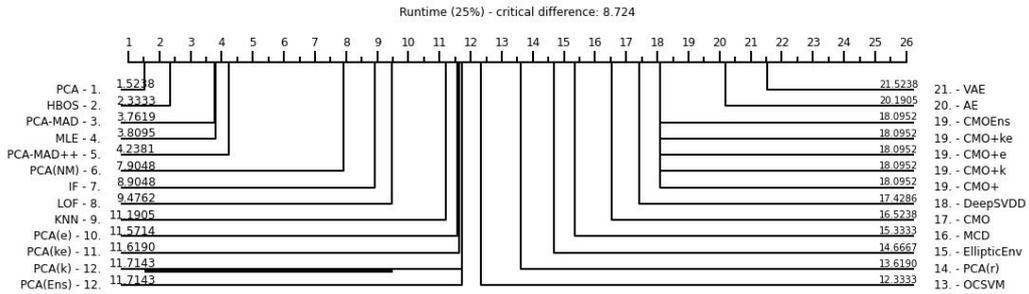
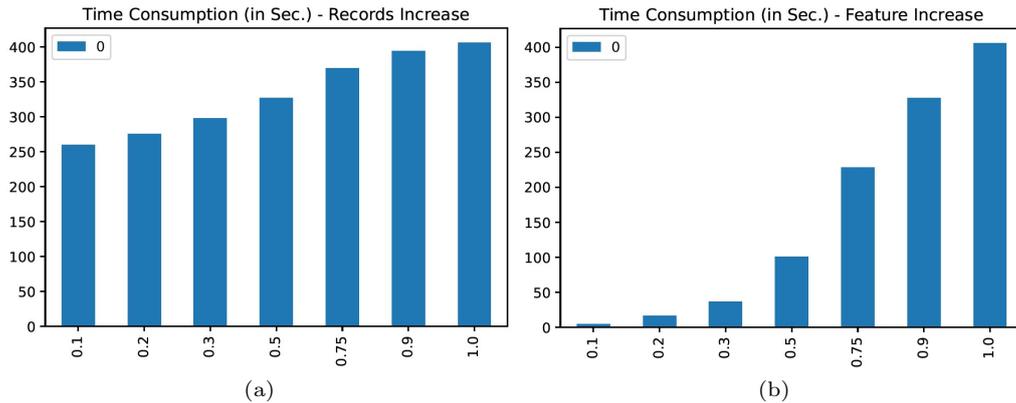
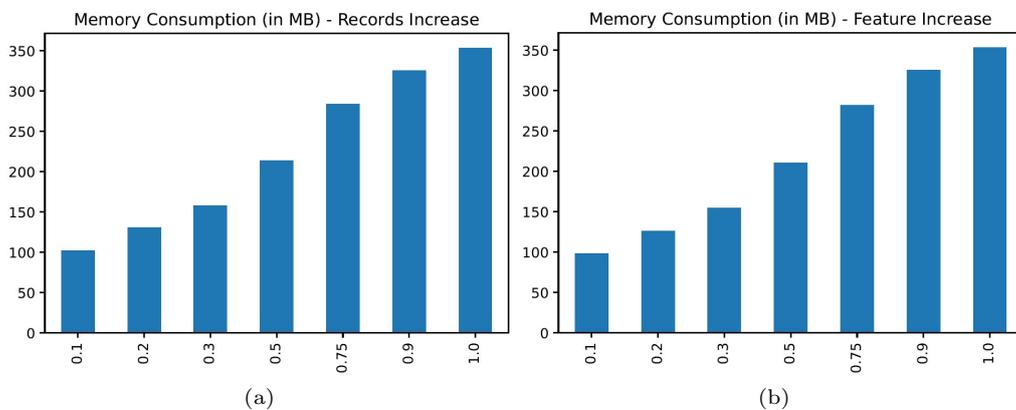


Fig. 15: Average Ranks of Runtime performances based on the top 25% of all possible principal components. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

4.2.4 Memory and Runtime Dependencies

In order to address the dependencies of CMO* variants related to its memory and runtime consumption behavior we made the following investigations. We used with 20newspaper the largest among all compared datasets (11097 samples, 768 features, 222.8MB raw size) and increased the fraction of samples or features respectively from 0.1 up to 1.0 in each of our tests. In Fig. 16 one can observe that in (a) the time consumption is only slightly but constantly increasing with $O(\log n)$ by an increasing amount of records, while in (b) the increase of features shows a non-linearly increasing runtime of $O(n^2)$. For the second experiment investigating in memory dependency, one can observe in Fig. 17 a) and b) that there is a nearly linear increase of $O(n)$ in memory consumption for both records and feature increase.

24 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD***Fig. 16:** CMO* time consumption dependent on records and features.**Fig. 17:** CMO* memory consumption dependent on records and features.

4.2.5 Result Discussion

The evidence for the contribution of CoMadOut specific *scoring* of inlier- and outlier-regions as part of the CMO approaches is investigated by excluding the outlier-resistant comedian matrix of coMAD-PCA and replacing it with the outlier-sensitive covariance matrix of standard PCA, both done with variants PCA(NM), PCA(r), PCA(k), PCA(e), PCA(ke) and PCA(Ens). Comparing standard PCA based outlier scoring with CMO-based outlier scoring one observes for the latter that PCA(Ens) shows better AUPRC performances (3/3) than PCA (9/9) on all principal components (100%) but also on the heavily reduced top 25% principal components in terms of (average) ranks. However, the significance tests in Fig. 6 and Fig. 9 as well as the comparable average performances show that only the combination of both, outlier-robust dimensionality reduction and CoMadOut scoring methods, allow permanently better performance scores (always with at least one CMO variant ahead) than standard PCA based methods without median usage.

The evidence for the contribution of outlier scoring based on our nearly unmodified comedian matrices is shown by the CoMadOut variant CMO+ which shows together with the kurtosis-weighted variant CMO+k competitive

ranks compared to the majority of all other outlier detection algorithms in terms of AP, AUROC, AUPRC and P@n. The evidence for the contribution of the distribution tailedness based outlier score weighting is shown when comparing related CoMadOut variant CMO+k with the most related variance- and z-score-weighted approaches of PCA-MAD. In terms of average precision (AP) CMO+k outperforms the related PCA-MAD approaches and shows a competitive overall performance also in terms of AUPRC. However, despite considerable performances of CMO+ and CMO+e, both cannot compete with CMO+k since they neglect to include the kurtosis-based weighting according to the datasets tailedness.

The evidence for robustness is especially shown when comparing CoMadOut variants with outlier-robust methods like PCA-MAD, MCD, etc. on the one side and outlier-sensitive methods like PCA, MLE, etc. on the other side on datasets with a high percentage of outliers and noise (arrhythmia, breastw, pima, satellite, boston). On the majority of these datasets CoMadOut shows compared to the outlier-sensitive methods and partially even also to robust outlier detection methods a better performance in terms of AP, AUROC and AUPRC. Furthermore, the difference is directly visible when comparing the test dataset plots of the principal components of coMAD-PCA and standard PCA next to each other where those of standard PCA are sensitive towards the shown outlier (cf. Fig. 2 and result tables for details).

Beside the strengths of our approaches, they also have limitations. On datasets with clear borders between normal and abnormal most CMO variants cannot benefit of its weight-based scoring between normal and abnormal. According to EDA, CMO shows better performances when there are less extreme overlaps within normal and abnormal distributions on the principal components. In the experiments CMO variants showed the best performance on datasets arrhythmia, annthyroid, breastw, thyroid, pendigits, musk, satimage-2 and boston. In addition to that, a restricted variation among the tailednesses of the distributions leads to less effectiveness of variants CMO+k, CMO+e and CMO+ke, whereas reduced noise instance do not allow comedian PCA to show its strength for outlier resistant eigenvectors. A further weakness which has to be paid for fast execution times are the numerical calculations on the whole dataset instead of batch-wise computation leading to high memory consumption and with that to restrictions for hardware and application domains. In addition to that, also the following property identified by [Sajesh and Srinivasan](#) also hold for our approach due to the comparable comedian matrix basis. In this scope he states the non-positive-semi-definiteness of the comedian matrix. Furthermore, he found breakdown values between outlier ratios $\alpha \in [10, 45]$ and identified false detection rates of 0.1% for symmetrically distributed outliers and 0.7% for asymmetrically distributed outliers, which need to be also investigated for our approach in future work. Further potential for optimization is the loss of performance of CMO variants which comes along with the reduction of principal components. In this case, CMO+k shows with stable ranks the most robust performances among all other CMO variants.

Table 2: Parameters, ranges and stepsizes used for parameter tuning.

algorithm or group	parameter	range	stepsize
PCA-based (CMO*, PCA*)	n_components	[0.25, 1.0]	0.25
Nearest neighbor based (LOF, KNN)	n_neighbors	[5, 100]	5
Tree-based (IF)	n_estimators	[10, 100]	10
HBOS	n_bins	[10, 100]	10
OCSVM	nu	(0, 1)	0.1
MCD, EllipticEnv	support_fraction	(0, 1]	0.1
AE, VAE, DeepSVDD	batch_size	2^n with $n \in [2,8]$	1

4.3 Results based on Parameter-Tuning

Although it is not common for an unsupervised setting like ours since one may not have ground truth labels available, we utilize the ground truth labels to address potential performance improvements due to model- and hyperparameter tuning. Therefore and to provide an alternative for commonly used default parameters, we conduct a dataset specific optimization by grid search for all algorithms as far as they offer a sensitive tuning parameter. In this scope we search with seed 0 and the following parameters, value ranges and step sizes listed in Table 2.

We optimize the algorithms according to Average Precision (AP) and follow with that metric the approach of (Zhao, Rossi, & Akoglu, 2021). Accordingly, the best dataset specific algorithm parameters are provided within Table A16. These by AP selected parameters are the basis for the conducted tests which report the dataset specific algorithm performances by AUROC (cf. Table A11), by AP (cf. Table A12), by AUPRC (cf. Table A13), by P@n (cf. Table A14) and by Runtime (cf. Table A15). As already done for the principal component ratio dependent performance analysis we also summarize the algorithm performances by their achieved average rank over all datasets and for each performance metric by critical difference plots, which can be found for AUROC (cf. Figure 18), for AP (cf. Figure 19), for AUPRC (cf. Figure 20), for P@n (cf. Figure 21) and for Runtime (cf. Figure 22).

Compared to the previous analysis KNN shows superior performance in terms of AUROC, AP, AUPRC and P@n. Moreover our CMO+k demonstrates stable on par performances being four times among the top 5 algorithms. Also our CMO+ performs comparably stable being not worse than rank 7. As already demonstrated in earlier analysis, one weakpoint of our CMO* algorithms are worse runtimes which remain as expected also after parameter tuning. In addition to that, our ensemble variant CMOEns still shows good performance for AUPRC. Besides KNN also the algorithm MCD performs comparably well achieving ranks not worse than rank 6 for AUROC, AP, AUPRC and P@n.

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD 27

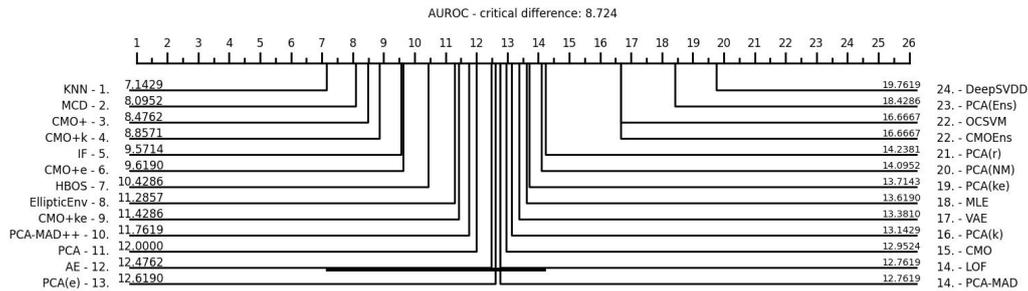


Fig. 18: Average Ranks of AUROC performances based on parameter tuning. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

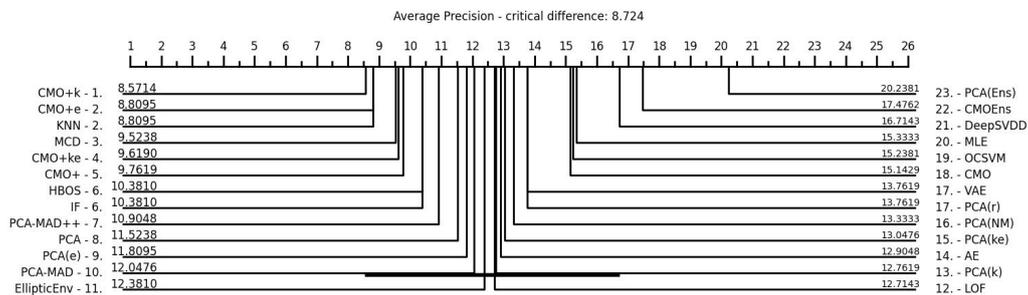


Fig. 19: Average Ranks of Average Precision performances based on parameter tuning. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

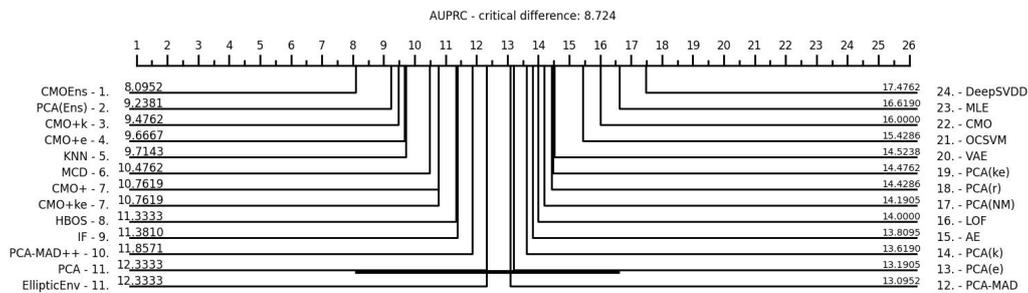


Fig. 20: Average Ranks of AUPRC performances based on parameter tuning. Algorithms within bold black bars show no significant better performance. CMO* are ours, 1. is the best.

5 Conclusion

In this work we proposed CoMadOut, a robust outlier detection algorithm using comedian PCA in order to apply PCA without sensitivity to outliers providing an initially selective inlier region besides an extending inlier

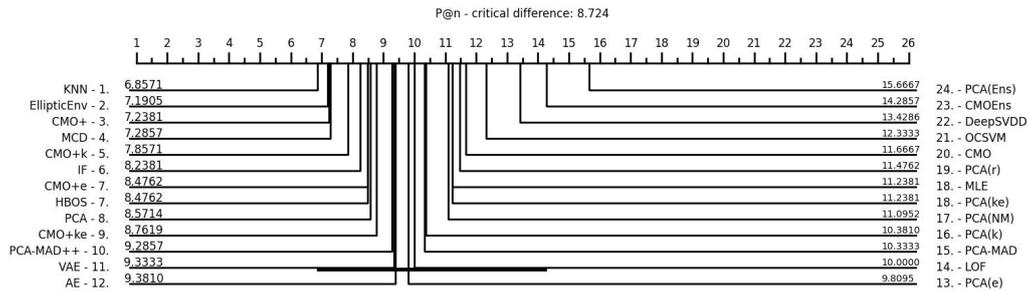
28 *CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD*

Fig. 21: Average Ranks of $P@n$ performances based on parameter tuning. Algorithms within bold black bars show no significant better performance. CMO^* are ours, 1. is the best.

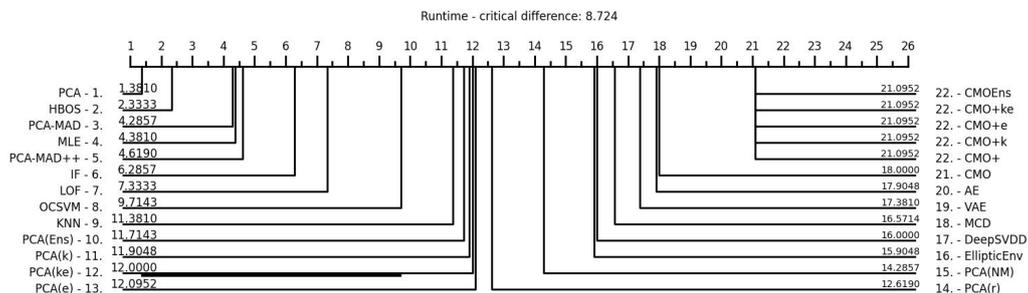


Fig. 22: Average Ranks of runtime performances based on parameter tuning. Algorithms within bold black bars show no significant better performance. CMO^* are ours, 1. is the best.

noise margin by measures of in-distribution (ID) and measures of out-of-distribution (OOD). They care for distribution based alignment of the outlier scores of each principal component, and with that, for an appropriate alignment of the decision boundary between normal and abnormal instances. Based on the assumption that samples are only considered as outliers when they are not part of our enhanced inlier region or when they exceed the distribution tailedness weighted outlier score boundary, we demonstrated an on par performance to various kinds of (robust) state-of-the-art outlier detection methods. The consideration of model selection by meta-learners (Zhao et al., 2021), of estimating the ideal number of principal components, investigating the performance loss of achieving positive-semi-definiteness while preserving the properties of the original comedian matrix and runtime optimization for CoMadOut in a future work could make our approach even more applicable to high-dimensional datasets and thereby also beneficial to other application domains with the requirement of robust outlier detection.

Author Contributions

- Conceptualization: ALO (Lead), DKA (Supporting), PKR (Supporting)
- Methodology: ALO (Lead), DKA (Supporting)
- Formal Analysis and Investigation: ALO (Lead)
- Implementation: ALO (Lead)
- Writing – original draft: ALO (Lead), DKA (Supporting), MAH (Supporting), PKR (Supporting)
- Writing – review & editing: ALO (Lead), DKA (Supporting), MAH (Supporting), PKR (Supporting)
- Funding acquisition: PKR (Lead)

Funding

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

Declarations

- Employment, Financial interests, Non-Financial interests - Not applicable
- Conflict of interest - No conflicts or competing interests to declare
- Ethics approval - Not applicable
- Consent to participate/publication - Not applicable (all agree)
- Employment, Financial interests, Non-Financial interests - Not applicable
- Availability of data and materials - Not applicable (publicly available)
- Code availability - planned at <https://github.com/lohrera/CoMadOut>

Appendix A Experiment Result Tables

30

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD

Table A1: Average Precision (AP) performance of 6 CoMadOut (CMO*) variants on the left compared to 20 other anomaly detection algorithms on the right on 21 real world datasets. The results are based on 100% of all possible principal components.

	CMO	CMO+k	CMO+e	CMO+ke	CMO+Em	MAD++	PCA-MAD	HBOS	IF	PCA	PCA(NM)	PCA(G)	PCA(k)	PCA(e)	PCA(e)	PCA(Em)	LOF	KNN	OCSVM	MCD	EllipticEnv	MLE	AE	VAE	DeepSVDD	AVG		
2news	0.044724	0.045308	0.045306	0.046634	0.046287	0.050000	0.045707	0.045711	0.047568	0.045942	0.046790	0.047336	0.046875	0.056310	0.052243	0.048562	0.049102	0.047434	0.045803	0.047201	0.050000	0.045824	0.046800	0.046733	0.048054	0.047372		
arthritis	0.63627	0.623570	0.601388	0.431523	0.436417	0.311826	0.450689	0.480558	0.419830	0.471969	0.356987	0.440065	0.343874	0.342964	0.207777	0.219795	0.202972	0.414867	0.472525	0.136475	0.232062	0.158957	0.285905	0.397805	0.397781	0.169328	0.360928	
cardio	0.251333	0.377497	0.266436	0.397465	0.393873	0.201126	0.677366	0.567422	0.363401	0.612072	0.339740	0.374698	0.343690	0.218981	0.203313	0.154308	0.285894	0.533209	0.337562	0.355416	0.463834	0.626168	0.610493	0.13230	0.394006			
shuttle	0.077981	0.025916	0.662649	0.635331	0.642549	0.270832	0.184498	0.153584	0.183403	0.315029	0.191271	0.074167	0.077598	0.056714	0.077280	0.070682	0.070823	0.203271	0.219291	0.105847	0.048815	0.091504	0.117524	0.231436	0.191529	0.112233	0.237947	
breastw	0.910694	0.987017	0.961325	0.966530	0.863462	0.777642	0.910820	0.957041	0.959415	0.972006	0.957770	0.973123	0.990672	0.975286	0.935219	0.855334	0.718703	0.271975	0.296932	0.940377	0.968966	0.968966	0.939455	0.847350	0.938759	0.833327	0.890195	
letter	0.068543	0.081984	0.106329	0.118586	0.118100	0.081380	0.152126	0.078427	0.077260	0.094046	0.076354	0.094515	0.064677	0.072535	0.073068	0.078461	0.059630	0.486379	0.306940	0.388391	0.179835	0.179835	0.226600	0.098940	0.076728	0.101201	0.139457	
thyroid	0.063508	0.794024	0.793795	0.770701	0.772753	0.524549	0.097451	0.315292	0.452138	0.318210	0.355975	0.024655	0.021994	0.033335	0.281724	0.323485	0.061763	0.110424	0.251382	0.149337	0.702109	0.077903	0.257805	0.466601	0.355674	0.112277	0.334312	
mnist	0.23094	0.067335	0.095323	0.024553	0.024458	0.043108	0.150612	0.159692	0.127278	0.109965	0.198108	0.105948	0.213702	0.234845	0.178996	0.199834	0.077604	0.089125	0.157636	0.183394	0.05255	0.033931	0.177659	0.161727	0.198468	0.156744	0.135337	
pinas	0.437570	0.498754	0.513239	0.507109	0.509339	0.383039	0.477193	0.448181	0.510109	0.500960	0.466714	0.535907	0.547032	0.528595	0.542989	0.524839	0.375802	0.368692	0.658835	0.373526	0.488681	0.488681	0.496046	0.412383	0.427524	0.478584	0.473012	
mnist	0.095039	0.281436	0.223809	0.057856	0.054205	0.890963	1.000000	0.968157	0.976955	0.999789	0.774522	0.059299	0.972378	0.027482	0.028949	0.082694	0.027626	0.023741	0.041231	0.939350	0.221350	0.897909	1.000000	0.000000	0.999831	0.044309	0.49129	
opendigits	0.032639	0.046866	0.082706	0.053972	0.053941	0.033872	0.021070	0.029450	0.149819	0.053904	0.026683	0.033240	0.018456	0.020115	0.037178	0.033185	0.026926	0.064621	0.030085	0.028037	0.021865	0.024264	0.026518	0.026608	0.026621	0.028201	0.037655	
pendigits	0.073021	0.196588	0.118785	0.080069	0.084143	0.080811	0.099740	0.107373	0.350329	0.218894	0.218333	0.038383	0.021402	0.024763	0.020515	0.025568	0.021097	0.036922	0.064362	0.207590	0.069130	0.069130	0.069130	0.055515	0.233789	0.221963	0.034301	0.107382
mnist	0.154220	0.296131	0.396662	0.196696	0.197947	0.194243	0.092369	0.092069	0.156714	0.369967	0.382761	0.104259	0.108771	0.175391	0.063674	0.063638	0.099170	0.184651	0.338316	0.092069	0.317719	0.093366	0.381626	0.397418	0.384055	0.098302	0.205293	
shuttle	0.817244	0.819616	0.141191	0.398526	0.171439	0.801167	0.131533	0.233759	0.325127	0.977436	0.915382	0.690709	0.771436	0.246877	0.093457	0.093457	0.093457	0.102438	0.175984	0.187763	0.841147	0.071511	0.871268	0.914748	0.915435	0.748381	0.523833	
satellite	0.608460	0.647049	0.600381	0.658318	0.647904	0.539910	0.654204	0.616250	0.693566	0.656724	0.605734	0.540718	0.488460	0.596278	0.384551	0.298822	0.377980	0.371941	0.531129	0.308323	0.767194	0.767194	0.514233	0.691912	0.605760	0.256823	0.564378	
satimage-2	0.734994	0.944606	0.972979	0.673542	0.763823	0.102453	0.969029	0.968962	0.710239	0.924362	0.872453	0.932183	0.831035	0.965457	0.149419	0.214721	0.059414	0.032237	0.341379	0.016451	0.684942	0.684942	0.397157	0.829547	0.871151	0.012272	0.603042	
wine	0.141460	0.251807	0.320308	0.178232	0.183975	0.163256	0.393421	0.990909	0.165327	0.212801	0.255486	0.990909	0.990909	0.990909	0.990909	0.990909	0.312500	0.969798	0.954040	0.096980	0.729536	0.729536	0.112100	0.154926	0.099019	0.302085	0.487375	
vowels	0.153119	0.098985	0.211464	0.062379	0.062380	0.069237	0.240038	0.127177	0.081309	0.171588	0.068347	0.174115	0.138590	0.322029	0.284294	0.272224	0.080879	0.310235	0.551346	0.105171	0.055191	0.055191	0.360970	0.079349	0.047384	0.110798	0.119007	
glass	0.131993	0.082361	0.086750	0.126128	0.102578	0.048446	0.075337	0.095308	0.069160	0.106554	0.071012	0.246049	0.257409	0.339600	0.033112	0.032862	0.049314	0.153510	0.166910	0.087288	0.117569	0.091896	0.080438	0.077859	0.73194	0.098403	0.111310	
wic	0.429879	0.474648	0.444696	0.615533	0.603364	0.310185	0.440517	0.579639	0.702767	0.399963	0.556957	0.477076	0.514638	0.544237	0.502995	0.336312	0.234497	0.427263	0.302821	0.561504	0.433796	0.434206	0.434004	0.448531	0.515641	0.131142	0.479236	
beston	0.876855	0.970508	0.837795	0.779963	0.743719	0.570373	0.586431	0.578611	0.545064	0.733102	0.809674	0.562563	0.418150	0.408855	0.422201	0.419839	0.432925	0.559320	0.631631	0.351397	0.700888	0.357365	0.589993	0.809424	0.8578053	0.386419	0.616977	
WIN	1.000000	2.000000	2.000000	0.000000	0.000000	0.000000	2.000000	3.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	2.000000	2.000000	0.000000	0.000000	1.115885	
AVG	0.329487	0.435576	0.394927	0.396174	0.355993	0.268499	0.373990	0.418043	0.412730	0.467130	0.436372	0.390728	0.359419	0.383040	0.265462	0.271975	0.178824	0.250976	0.357955	0.239297	0.439758	0.285878	0.369682	0.426352	0.422152	0.208396	0.351608	
ARK	14.238905	9.806524	10.000000	10.000000	10.000000	11.019048	15.809524	12.571429	9.932381	14.952381	13.190476	13.333333	15.952381	15.23810	20.000000	14.952381	11.761905	16.190476	11.666667	15.32810	14.285714	11.571429	17.619048	12.919414	17.619048	12.919414		
RK	17.000000	2.000000	3.000000	3.000000	9.000000	20.000000	14.000000	7.000000	4.000000	1.000000	5.000000	12.000000	15.000000	13.000000	21.000000	21.000000	18.000000	15.000000	22.000000	10.000000	10.000000	10.000000	16.000000	6.000000	6.000000	8.000000	23.000000	12.346415

Table A3: AUROC performance of 6 CoMadOut (CMO*) variants on the left side compared to 20 other anomaly detection algorithms on the right side on 21 real world datasets. The results are based on 100% of all possible principal components.

	CMO	CMO+	CMO+k	CMO++	CMO+ke	CMOEmS	MAD++	PCA-MAD	PCA-MAD++	PCA-MADEmS	IF	PCA	PCA(NM)	PCA(t)	PCA(r)	PCA(c)	PCA(ke)	PCA(EmS)	LOF	KNN	OCSVM	MCD	EllipticEm	MILE	AE	VAE	DeepSVDD	AVG
20news	0.451249	0.445444	0.446216	0.461179	0.459844	0.496761	0.448359	0.465799	0.463108	0.457231	0.471417	0.458050	0.475428	0.472949	0.532670	0.525842	0.470352	0.494002	0.472295	0.468315	0.453092	0.499595	0.448672	0.471880	0.471177	0.488892	0.472982	
arthritis	0.836199	0.839771	0.808447	0.722131	0.716635	0.715732	0.810000	0.813300	0.810606	0.806979	0.774808	0.755613	0.581449	0.616541	0.870191	0.587613	0.587533	0.788307	0.800447	0.809047	0.660228	0.507576	0.751185	0.770649	0.777328	0.518704	0.706781	
cardio	0.809043	0.900412	0.839776	0.772895	0.762637	0.666299	0.937000	0.955000	0.852541	0.931708	0.950577	0.814162	0.938777	0.834084	0.670739	0.820328	0.545784	0.686149	0.935234	0.774689	0.806744	0.806454	0.947740	0.950095	0.496966	0.814101		
santhyroid	0.546527	0.948366	0.959160	0.952467	0.955903	0.638354	0.903000	0.878000	0.624088	0.827367	0.672882	0.500000	0.513267	0.397309	0.544121	0.513324	0.469515	0.739194	0.755748	0.546726	0.918901	0.598363	0.641487	0.705894	0.674390	0.491677	0.685874	
breastw	0.974829	0.992290	0.985318	0.984003	0.902677	0.844547	0.989000	0.990000	0.985129	0.987608	0.958941	0.984328	0.994921	0.987862	0.964539	0.934430	0.857759	0.371000	0.975301	0.978698	0.988281	0.988281	0.972389	0.866450	0.893671	0.892800	0.932540	
letter	0.558680	0.593180	0.625427	0.697177	0.697053	0.599667	0.659000	0.655000	0.580307	0.643549	0.522947	0.614420	0.497160	0.531627	0.589740	0.611647	0.470000	0.899793	0.910880	0.889633	0.807140	0.807140	0.800093	0.574923	0.514597	0.493317	0.647946	
thyroid	0.811797	0.909294	0.909285	0.682485	0.694970	0.685485	0.555791	0.713000	0.713000	0.685784	0.670425	0.632157	0.719612	0.742119	0.724575	0.711833	0.721545	0.538890	0.538848	0.607688	0.539302	0.680888	0.680888	0.674448	0.561894	0.581094	0.629758	0.653529
mammography	0.892107	0.801632	0.804133	0.574480	0.560648	0.651028	0.884000	0.884000	0.829923	0.835810	0.886336	0.810210	0.881384	0.874721	0.821467	0.832637	0.749375	0.719300	0.837803	0.872083	0.614311	0.648367	0.806201	0.762918	0.887052	0.652348	0.786549	
pinna	0.600534	0.684858	0.682485	0.694970	0.685485	0.555791	0.713000	0.713000	0.685784	0.670425	0.632157	0.719612	0.742119	0.724575	0.711833	0.721545	0.538890	0.538848	0.607688	0.539302	0.680888	0.680888	0.674448	0.561894	0.581094	0.629758	0.653529	
msk	0.776391	0.885781	0.683754	0.757678	0.739659	0.766689	1.000000	1.000000	0.998114	0.999181	0.999993	0.990456	0.732550	0.908981	0.468003	0.440690	0.818550	0.303842	0.572822	0.999977	0.507938	0.997072	1.000000	0.999994	0.502692	0.786643		
optdigits	0.587071	0.711016	0.692935	0.753216	0.746380	0.562012	0.717000	0.386000	0.870159	0.709890	0.512175	0.435119	0.288000	0.338147	0.639362	0.589550	0.457235	0.588179	0.435749	0.639666	0.406535	0.445129	0.509045	0.509964	0.511583	0.491563	0.357562	
pendigits	0.846751	0.922988	0.885689	0.850376	0.838766	0.861025	0.889000	0.900000	0.928259	0.940442	0.935488	0.561010	0.517445	0.568327	0.491711	0.548374	0.443565	0.479379	0.708691	0.935419	0.839040	0.839040	0.762802	0.983148	0.936742	0.511758	0.763371	
mnist	0.656727	0.796156	0.846778	0.727543	0.724774	0.760470	0.905000	0.910000	0.686770	0.801169	0.849893	0.498743	0.486077	0.618767	0.328469	0.328101	0.535589	0.644880	0.794059	0.500000	0.810596	0.500671	0.846204	0.863972	0.851149	0.494791	0.683654	
shuttle	0.982007	0.992233	0.585038	0.989627	0.733687	0.975770	0.997000	0.998000	0.979293	0.987123	0.989773	0.975450	0.985201	0.763447	0.232161	0.478280	0.444971	0.554276	0.703913	0.724879	0.990118	0.499989	0.982406	0.992753	0.989824	0.910308	0.824905	
satellite	0.711555	0.656244	0.679761	0.727751	0.723097	0.685416	0.784000	0.783000	0.765947	0.696156	0.601161	0.580906	0.537131	0.628290	0.491541	0.525934	0.598534	0.539355	0.670037	0.487910	0.803192	0.803192	0.639451	0.723482	0.601224	0.340105	0.645416	
satimage-2	0.991061	0.997899	0.999206	0.994801	0.996386	0.945743	0.999000	0.999000	0.971605	0.983823	0.977168	0.996790	0.990078	0.998811	0.969733	0.958673	0.901954	0.532484	0.929627	0.420907	0.995382	0.995382	0.983800	0.984984	0.977271	0.308312	0.913490	
wine	0.726891	0.863866	0.910084	0.755462	0.763866	0.732353	0.938000	0.940000	0.766387	0.810168	0.821008	0.999160	0.999160	0.999160	0.999160	0.999160	0.907563	0.997479	0.996218	0.475210	0.975966	0.975966	0.646680	0.681849	0.530420	0.726697	0.843647	
vowels	0.827752	0.726873	0.823073	0.691821	0.691707	0.721707	0.838000	0.838000	0.676572	0.771377	0.606230	0.884054	0.736415	0.850569	0.905292	0.898471	0.765806	0.942973	0.970799	0.778336	0.702404	0.702404	0.912991	0.665112	0.610310	0.567991	0.779436	
glass	0.738496	0.728455	0.730081	0.816260	0.796206	0.554472	0.740000	0.728000	0.671003	0.702331	0.604336	0.803794	0.515989	0.798374	0.337127	0.327371	0.550350	0.832320	0.864499	0.393496	0.800434	0.775610	0.585799	0.617778	0.562927	0.638171	0.672034	
wbc	0.931796	0.943944	0.934374	0.932373	0.923569	0.898123	0.938000	0.938000	0.938000	0.935307	0.938575	0.938942	0.940046	0.925304	0.940510	0.873850	0.929705	0.909980	0.938242	0.917647	0.917727	0.939772	0.876837	0.895505	0.58135	0.909928		
boston	0.923735	0.981515	0.894999	0.822288	0.787129	0.667598	0.666619	0.623272	0.674867	0.842738	0.922625	0.622908	0.606683	0.591837	0.619958	0.612714	0.617634	0.663643	0.726582	0.497113	0.739883	0.502793	0.734406	0.841443	0.912070	0.511530	0.715449	
WIN	1.000000	3.000000	2.000000	0.000000	0.000000	0.000000	1.000000	4.000000	2.000000	1.000000	1.000000	1.000000	2.000000	1.000000	2.000000	1.000000	1.000000	3.000000	3.000000	3.000000	1.000000	1.000000	1.000000	1.000000	0.000000	0.000000	1.076923	
AVG	0.774100	0.829682	0.796649	0.793721	0.771089	0.717469	0.843951	0.857527	0.796892	0.826366	0.789475	0.735319	0.622233	0.705859	0.621445	0.665945	0.650760	0.673243	0.764515	0.665191	0.796522	0.690889	0.780212	0.779641	0.765963	0.554579	0.749928	
ARK	12.238905	8.571429	9.952381	10.666667	12.476100	16.095238	6.809524	6.428571	10.761905	8.666667	11.190476	13.333333	14.523810	12.714286	16.000000	15.333333	19.190476	14.857143	11.142857	14.857143	12.806524	12.142857	12.333333	20.857143	12.761905			
HK	12.000000	3.000000	6.000000	14.000000	22.000000	2.000000	1.000000	7.000000	4.000000	9.000000	10.000000	15.000000	18.000000	15.000000	21.000000	20.000000	24.000000	10.000000	10.000000	8.000000	8.000000	19.000000	16.000000	13.000000	25.000000	13.236769		

Table A9: Runtime in seconds of 6 CoMadOut (CMO*) variants on the left side compared to 20 other anomaly detection algorithms on the right side on 21 real world datasets. The results are based on 100% of all possible principal components.

	CMO+	CMO+h	CMO+e	CMO+ke	CMO+Ens	MAD++	PCA-	PCA-	MAD	PCA-	IF	PCA	PCA(NM)	PCA(r)	PCA(k)	PCA(e)	PCA(ke)	PCA(Ens)	LOF	KNN	OCSVM	MCD	EllipticEw	MLE	AE	VAE	DeepSVDD	AVG			
20news	215.852400	77.197400	77.197400	77.197400	77.197400	0.672800	0.626800	0.367400	1.252400	0.782400	0.3902800	40.512400	40.419200	40.149600	2.089200	2.997600	0.871000	1.45.237800	1.48.041400	0.395600	8.374400	10.467200	1.851000	50.551323	1.851000	50.551323					
arrhythmia	12.865400	7.719000	7.719000	7.719000	7.719000	0.171400	0.167600	0.039800	0.284000	0.052600	8.344800	3.025400	3.081200	3.032600	0.129800	0.151000	0.079000	0.129800	0.151000	0.079000	3.561400	4.984800	3.561400	0.171080	3.161200	4.896400	1.179000	3.612800			
cardio	1.699600	2.896200	2.896200	2.896200	2.896200	0.026800	0.019200	0.005000	0.284200	0.006600	4.22200	1.594400	1.584400	1.577200	1.585800	1.579400	1.579400	1.585800	1.579400	0.237000	0.956200	0.956200	0.956200	0.956200	0.956200	1.905200	2.017885				
anahydroid	1.486000	2.142800	2.142800	2.142800	2.142800	0.009200	0.007400	0.003600	0.524400	0.005200	0.328600	0.912200	0.908300	0.897600	0.885600	0.903600	0.885600	0.903600	0.885600	0.903600	3.005600	1.285400	1.126600	0.148000	21.275000	36.207200	4.616400	3.321289			
breastw	1.105400	2.324000	2.324000	2.324000	2.324000	0.006200	0.005000	0.002600	0.201200	0.002000	0.061400	1.059800	1.044200	1.046000	1.039000	0.033200	0.039800	0.033200	0.039800	0.033200	0.033200	0.033200	0.033200	0.033200	0.033200	3.998200	6.059800	1.335800	1.170685		
letter	1.905800	3.002000	3.002000	3.002000	3.002000	0.051000	0.039200	0.007600	0.287400	0.010400	0.687800	1.592200	1.616000	1.609000	1.626800	1.612200	1.614500	1.626800	1.612200	1.614500	0.239800	2.426800	2.426800	2.426800	2.426800	2.426800	5.571000	9.047200	1.399000	1.805908	
thyroid	1.289400	2.193400	2.193400	2.193400	2.193400	0.007600	0.006000	0.003600	0.335800	0.003600	0.180400	0.915600	0.896000	0.875200	0.211400	0.327200	0.815000	0.796400	0.815000	0.796400	0.815000	0.796400	0.815000	0.796400	0.815000	0.796400	0.815000	26.324200	2.893600	2.604469	
mnist	1.729200	2.259000	2.259000	2.259000	2.259000	0.030400	0.018600	0.005200	0.721400	0.008400	0.555000	0.951600	0.937000	0.935200	0.941000	0.957000	0.941000	0.957000	0.941000	0.957000	7.485200	1.783800	1.447000	0.361000	35.217000	56.142600	6.757800	5.021915			
pinna	1.108800	2.286600	2.286600	2.286600	2.286600	0.006000	0.004800	0.002000	0.208200	0.002000	0.062400	1.007000	0.977000	0.963400	0.993400	0.980400	0.014600	0.044400	0.014600	0.044400	0.083600	0.350200	0.335200	0.009200	4.351000	6.598800	1.351200	1.189500			
mnst	25.356000	7.744800	7.744800	7.744800	7.744800	0.194200	0.176200	0.046400	0.754200	0.093800	22.050400	4.684400	4.659200	4.645400	4.681400	4.580600	2.114600	1.860000	2.114600	1.860000	0.970000	139.778800	106.580200	0.330400	15.980600	21.878800	2.597800	15.79400			
opdigits	9.014800	3.971600	3.971600	3.971600	3.971600	0.110400	0.126600	0.027400	0.712000	0.044800	7.598000	2.429200	2.413600	2.374800	2.388200	3.379000	3.790400	3.379000	3.790400	3.379000	3.828800	0.933400	0.669600	0.147000	27.798000	36.596400	4.201400	5.613600			
pendigits	2.419000	3.030600	3.030600	3.030600	3.030600	0.052000	0.035600	0.010400	0.638400	0.013600	1.192200	1.673200	1.702600	1.694200	1.690000	1.650600	0.945000	1.224600	0.945000	1.224600	2.769200	4.896600	3.548800	0.164600	33.297600	47.038000	4.582200	4.858208			
mnst	24.529800	6.625800	6.625800	6.625800	6.625800	0.156800	0.139600	0.043400	1.296400	0.100400	22.498000	4.672200	4.668800	4.616200	4.579800	4.616200	11.332800	11.829000	11.332800	11.829000	10.829200	15.919400	13.922800	0.436200	34.911000	52.147600	4.974600	10.022469			
shuttle	5.285000	2.925400	2.925400	2.925400	2.925400	0.085400	0.088600	0.018400	3.024800	0.025600	4.220200	1.507600	1.524000	1.487600	1.493600	1.531200	35.354400	36.634200	35.354400	36.634200	24.644800	19.679200	12.121200	0.713000	215.443400	339.291400	26.681000	37.430231			
satellite	4.708000	3.480800	3.480800	3.480800	3.480800	0.052800	0.051800	0.015000	0.647800	0.025800	3.440000	1.974600	1.955600	1.980800	2.003400	1.951800	1.430600	1.661600	1.430600	1.661600	6.125000	12.182600	9.055400	0.166800	31.433600	43.564400	4.427800	5.625708			
satimage-2	4.293400	3.379400	3.379400	3.379400	3.379400	0.057400	0.053400	0.013800	0.594000	0.024800	3.698800	1.938600	1.904600	1.892800	1.927600	1.936800	1.929400	1.533200	1.929400	1.533200	4.967400	15.565400	18.960200	0.112000	26.516400	40.184600	4.067800	5.650609			
wine	1.147400	2.665200	2.665200	2.665200	2.665200	0.007000	0.006000	0.003400	0.165400	0.001800	0.030200	1.365000	1.325800	1.313400	1.347600	0.003800	0.007400	0.003800	0.007400	0.003800	0.003800	0.045000	0.037000	0.001200	1.366200	1.901200	1.002000	0.949054			
vowels	1.328000	2.622400	2.622400	2.622400	2.622400	0.009600	0.007400	0.003400	0.246800	0.003200	0.186600	1.283800	1.275000	1.256600	1.247800	1.244000	0.057200	0.110600	0.057200	0.110600	1.422000	0.936600	0.677000	0.012800	8.227400	11.132800	1.696000	1.689723			
glass	1.142600	2.332800	2.332800	2.332800	2.332800	0.005600	0.005000	0.002600	0.164600	0.002000	0.027400	1.077600	1.028400	1.042600	1.035400	1.066600	0.004200	0.012000	0.004200	0.012000	0.004000	0.037600	0.040600	0.003800	1.715200	2.561600	1.008400	0.909492			
wtc	1.363400	2.985200	2.985200	2.985200	2.985200	0.036800	0.027600	0.005600	0.184200	0.005000	0.193000	1.594400	1.573200	1.569800	1.607200	1.574600	0.013000	0.025600	0.013000	0.025600	0.013800	0.126200	0.130000	0.003000	2.141600	3.192000	1.180600	1.211254			
beston	1.197200	2.817200	2.817200	2.817200	2.817200	0.007800	0.007000	0.003200	0.193400	0.002000	0.029000	1.411600	1.396600	1.382400	1.395600	1.356600	0.008200	0.028200	0.008200	0.028200	0.034200	0.530600	0.496600	0.009400	3.378800	4.747000	1.229000	1.268338			
WIN	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.646154	
AVG	15.278062	6.980743	6.980743	6.980743	6.980743	0.083676	0.077819	0.029952	0.606714	0.027905	12.446276	3.667571	3.625619	3.650790	3.656276	3.635419	2.912743	3.694114	2.912743	3.694114	13.856200	17.957238	15.768914	0.441590	24.352610	36.903619	3.853505	7.725482			
ARK	17.333333	17.904762	17.904762	17.904762	17.904762	1.428571	8.523810	1.904762	11.905294	14.380092	12.333333	13.333333	13.000000	7.523810	9.095238	11.047619	15.806052	4.571429	19.857143	11.047619	15.285714	14.668686	14.761905	15.285714	14.668686	15.285714	14.668686				
RK	19.000000	20.000000	20.000000	20.000000	20.000000	5.000000	3.000000	1.000000	7.000000	2.000000	10.000000	15.000000	12.000000	11.000000	14.000000	13.000000	6.000000	8.000000	6.000000	8.000000	9.000000	18.000000	16.000000	4.000000	21.000000	22.000000	17.000000	12.807692			

Table A11: AUROC of 6 CoMadOut (CMO*) variants on the left compared to 20 other anomaly detection algorithms on the right on 21 real world datasets. The results are based on parameter tuning.

	CMO	CMO+	CMO+k	CMO+e	CMO+le	CMOas	MAD++	PCA-	PCA-	PCA-	FCA	HOS	IF	FCA	PCA(SM)	PCA(f)	PCA(k)	PCA(c)	PCA(lb)	PCA(Elas)	LOF	KNN	OCSVM	MCD	EllipticEw	MLE	AE	VAE	DeepSVDD	AVG											
2news	0.47353	0.46187	0.46151	0.46398	0.46292	0.49676	0.47397	0.46556	0.46310	0.46087	0.47187	0.47209	0.48010	0.47294	0.48010	0.47294	0.53270	0.52582	0.49384	0.551106	0.47225	0.48854	0.48648	0.49950	0.44972	0.47132	0.46987	0.51643	0.48259												
arhythmia	0.83618	0.83971	0.81951	0.81488	0.81987	0.77532	0.80679	0.80679	0.81096	0.80127	0.75318	0.75815	0.81049	0.69927	0.68261	0.68170	0.62910	0.81637	0.57880	0.61671	0.50757	0.75185	0.78760	0.78745	0.52042	0.72640	0.84465	0.80042	0.86484	0.72895	0.85891	0.85939									
cardio	0.84465	0.80042	0.86484	0.72895	0.76263	0.66299	0.94687	0.95385	0.85251	0.92072	0.95077	0.81462	0.95456	0.83498	0.89544	0.81988	0.84925	0.85242	0.90922	0.94399	0.88579	0.88140	0.89645	0.95800	0.95750	0.48831	0.85932	0.59156	0.89687	0.89687	0.89687	0.89687									
amthyroid	0.59116	0.95087	0.96387	0.96089	0.971759	0.65501	0.67287	0.59153	0.88807	0.82256	0.76621	0.49459	0.51322	0.39730	0.55905	0.48745	0.49518	0.70685	0.72467	0.55520	0.93170	0.72245	0.64148	0.70873	0.69298	0.49905	0.69067	0.49905	0.69067	0.49905	0.69067	0.49905	0.69067								
breastw	0.97702	0.99320	0.98531	0.98403	0.90282	0.84457	0.94767	0.96368	0.98512	0.98692	0.96019	0.90078	0.96344	0.95910	0.98103	0.97152	0.88018	0.54991	0.99279	0.98183	0.99180	0.99108	0.99180	0.97289	0.84733	0.80336	0.95318	0.93961	0.95318	0.93961	0.95318	0.93961	0.95318	0.93961							
letter	0.52127	0.93180	0.62527	0.70450	0.68310	0.59667	0.73413	0.56713	0.65897	0.61070	0.50127	0.60567	0.69173	0.52193	0.57167	0.59180	0.57367	0.913967	0.91080	0.89633	0.81679	0.81679	0.89493	0.59070	0.46155	0.51647	0.64880	0.51647	0.64880	0.51647	0.64880	0.51647	0.64880	0.51647	0.64880						
thyroid	0.86879	0.99284	0.99270	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240	0.99240					
mnist	0.86165	0.85106	0.82760	0.67407	0.63135	0.74752	0.86491	0.83994	0.82923	0.86256	0.889312	0.80207	0.85281	0.83467	0.83582	0.82426	0.77221	0.80136	0.85215	0.82688	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966	0.84966				
pinna	0.59969	0.68485	0.68248	0.69497	0.68548	0.56937	0.64993	0.60702	0.65798	0.68642	0.63215	0.72643	0.72783	0.72457	0.736896	0.69264	0.55981	0.63217	0.64482	0.67448	0.70613	0.70613	0.67448	0.58576	0.59564	0.68739	0.65757	0.68739	0.65757	0.68739	0.65757	0.68739	0.65757	0.68739	0.65757	0.68739	0.65757				
mnist	0.77639	0.99180	1.00000	0.91801	0.98923	0.92415	1.00000	1.00000	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996	0.99996			
opendigits	0.77493	0.81478	0.80227	0.82580	0.81780	0.72452	0.74831	0.57926	0.880716	0.70980	0.51281	0.51829	0.52845	0.61087	0.51663	0.66590	0.52851	0.58808	0.54089	0.50854	0.41789	0.49930	0.50505	0.53227	0.46619	0.52896	0.62979	0.52896	0.62979	0.52896	0.62979	0.52896	0.62979	0.52896	0.62979	0.52896	0.62979	0.52896	0.62979		
pendigits	0.87107	0.94740	0.94801	0.87109	0.86754	0.86102	0.93478	0.91640	0.93510	0.83970	0.94580	0.64940	0.54273	0.57563	0.61185	0.39167	0.46636	0.52293	0.954046	0.86980	0.86979	0.76282	0.93205	0.93514	0.56184	0.78989	0.56184	0.78989	0.56184	0.78989	0.56184	0.78989	0.56184	0.78989	0.56184	0.78989	0.56184	0.78989			
mnist	0.65672	0.79615	0.86778	0.72743	0.72017	0.76047	0.85858	0.82238	0.68677	0.80184	0.85499	0.49873	0.48607	0.61877	0.61877	0.67987	0.61877	0.61877	0.61877	0.61877	0.81289	0.50053	0.84620	0.86054	0.884007	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553		
shuttle	0.97874	0.98515	0.58508	0.98927	0.73387	0.97287	0.60218	0.75718	0.98921	0.997233	0.99249	0.96070	0.98227	0.86504	0.98426	0.96767	0.97885	0.65261	0.81086	0.79067	0.99512	0.50012	0.98206	0.99295	0.99024	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553	0.98062	0.88553		
satellite	0.71308	0.65624	0.67976	0.72959	0.73476	0.65895	0.64734	0.67129	0.827605	0.70210	0.64616	0.58923	0.54520	0.64304	0.66938	0.68476	0.63705	0.57023	0.76187	0.51067	0.80450	0.80450	0.63641	0.71961	0.69278	0.54834	0.67321	0.54834	0.67321	0.54834	0.67321	0.54834	0.67321	0.54834	0.67321	0.54834	0.67321	0.54834	0.67321		
satimage-2	0.94324	0.99789	0.99206	0.99120	0.99907	0.94573	0.99818	0.99679	0.96187	0.98329	0.97161	0.99672	0.98460	0.99881	0.97162	0.98624	0.90195	0.95945	0.99893	0.60986	0.99906	0.99906	0.98380	0.98560	0.97727	0.93948	0.95224	0.93948	0.95224	0.93948	0.95224	0.93948	0.95224	0.93948	0.95224	0.93948	0.95224	0.93948	0.95224		
wine	0.74287	0.85964	0.92018	0.88672	0.90331	0.78273	0.95798	0.999160	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	0.86974	
vowels	0.82773	0.72637	0.82073	0.83462	0.80689	0.72107	0.80262	0.70385	0.68012	0.72805	0.66699	0.88808	0.73615	0.85059	0.95292	0.80452	0.76856	0.93874	0.970799	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	0.93529	
glass	0.79024	0.78398	0.78873	0.81734	0.74254	0.59874	0.51405	0.66231	0.67103	0.70386	0.57290	0.71205	0.69487	0.88235	0.84922	0.908943	0.64662	0.77019	0.86489	0.43446	0.78266	0.65181	0.68579	0.63197	0.66612	0.64192	0.69889	0.64192	0.69889	0.64192	0.69889	0.64192	0.69889	0.64192	0.69889	0.64192	0.69889	0.64192	0.69889	0.64192	0.69889
wbc	0.91703	0.94210	0.94790	0.92373	0.92569	0.80812	0.87175	0.94063	0.94579	0.92306	0.92170	0.93841	0.956383	0.95849	0.92704	0.90510	0.87395	0.94973	0.95151	0.94010	0.94143	0.94143	0.93077	0.89377	0.89276	0.89389	0.91669	0.89389	0.91669	0.89389	0.91669	0.89389	0.91669	0.89389	0.91669	0.89389	0.91669	0.89389	0.91669	0.89389	0.91669
boston	0.90697	0.981518	0.89499	0.80018	0.79418	0.67039	0.67159	0.62258	0.67486	0.84802	0.92265	0.70241	0.67919	0.65075	0.64531	0.61274	0.63957	0.71469	0.71419	0.55166	0.82300	0.82300	0.73406	0.91428	0.91724	0.45312	0.74482	0.45312	0.74482	0.45312	0.74482	0.45312	0.74482	0.45312	0.74482	0.45312	0.74482	0.45312	0.74482	0.45312	0.74482
VTN	0.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	
AVG	0.78478	0.845217	0.82883	0.82802	0.80831	0.74276	0.78920	0.78189	0.81847	0.82094	0.80944	0.76288	0.72334	0.73191	0.76958	0.79247	0.70437	0.75086	0.84962	0.71075	0.87737	0.76406	0.78621	0.78657	0.76523	0.62231	0.77450	0.62231	0.77450	0.62231	0.77450	0.62231	0.77450	0.62231	0.77450	0.62231	0.77450	0.62231	0.77450	0.62231	0.77450
RCP	12.95281	8.76190	8.57143	6.9048	11.42857	16.66667	11.76190	12.76190	10.42857	9.57142	12.00000	14.09228	14.23805	13.14285	18.42857																										

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD

Table A15: Runtime of 6 CoMadOut (CMO*) variants on the left compared to 20 other anomaly detection algorithms on the right on 21 real world datasets. The results are based on parameter tuning.

	CMO	CMO+	CMO+k	CMO+e	CMO+he	CMOEs	MAD++	PCA-MAD	PCA-MAD++	PCA-MAD	IF	PCA+PCA(NM)	PCA(c)	PCA(s)	PCA(c)	PCA(s)	PCA(Eus)	LOF	KNN	OCSVM	MCD	EllipticEnv	MLE	AE	VAE	DeepSVDD	AVG	
20news	199.937000	184.200600	184.200600	184.200600	184.200600	184.200600	1.515000	1.456000	0.979400	0.049600	0.347400	45.448300	4.708200	11.385200	42.129200	41.779400	25.580000	0.554400	1.762800	0.222800	130.954500	113.654000	0.464200	7.479400	3.162800	3.137800	0.162892	
arrhythmia	36.508000	32.706800	32.706800	32.706800	32.706800	32.706800	0.260400	0.265800	0.094200	0.191200	0.043000	17.815800	3.488200	2.575400	2.545600	0.186400	0.094800	0.320800	0.208400	0.094800	4.837600	5.117200	0.497600	1.684600	2.281600	1.747600	9.559000	
cardio	2.592400	11.937000	11.937000	11.937000	11.937000	11.937000	0.027600	0.022400	0.010600	0.363400	0.008400	1.304200	0.672400	1.163200	0.607200	0.996000	0.421800	0.334400	0.548000	0.421800	0.904200	0.860200	0.013600	3.101200	2.315200	1.759000	3.043855	
santhyroid	1.287000	7.613800	7.613800	7.613800	7.613800	7.613800	0.016000	0.012000	0.007000	0.051800	0.006200	1.069800	0.724400	0.732400	0.673000	0.670000	0.677200	1.205400	1.205400	0.662600	2.044600	2.549600	0.127200	4.407200	2.811000	3.193400	2.262846	
breastw	1.393400	9.923000	9.923000	9.923000	9.923000	9.923000	0.016200	0.013000	0.005000	0.155600	0.003200	0.056600	0.547000	0.502800	0.566200	0.571000	0.570600	0.029400	0.125600	0.050400	0.842800	0.778400	0.003600	2.541000	4.261800	2.564000	2.508177	
letter	1.618600	14.330800	14.330800	14.330800	14.330800	14.330800	0.048000	0.037600	0.016000	0.045600	0.008800	2.167600	1.432000	1.353000	1.349400	1.003800	0.302000	0.700000	0.292800	4.010400	5.324800	0.013600	0.606000	2.393600	2.044800	3.961915		
thyroid	1.409000	6.770800	6.770800	6.770800	6.770800	6.770800	0.030000	0.010200	0.005200	0.047600	0.005200	0.543800	0.677600	0.630400	0.640600	0.640600	0.503200	0.252400	0.736000	0.182200	1.385200	0.700200	0.038600	6.837600	2.472200	1.944400	2.058869	
mnist	2.046000	8.046200	8.046200	8.046200	8.046200	8.046200	0.021400	0.016000	0.007000	0.298800	0.009200	1.765000	0.694400	0.648000	0.647400	0.671400	0.592200	0.837600	2.351200	3.200800	1.482200	1.139200	0.284200	5.545400	2.718200	2.72154		
pinna	1.403600	9.373000	9.373000	9.373000	9.373000	9.373000	0.012200	0.012200	0.004000	0.081400	0.003200	0.177000	0.475800	0.449800	0.441200	0.513800	0.026200	0.132600	0.078800	0.830200	0.794600	0.003200	1.675800	3.403400	3.725000	2.375477		
mnst	81.628000	23.562800	23.562800	23.562800	23.562800	23.562800	0.219200	0.234400	0.057600	0.420800	0.134600	6.811600	4.510400	4.610800	4.311800	1.092000	0.401800	2.222000	3.163200	52.567200	55.270800	1.068200	2.161400	2.673600	3.088400	13.373962		
optdigits	9.227000	18.328200	18.328200	18.328200	18.328200	18.328200	0.106800	0.082200	0.043400	0.457600	0.048400	7.907600	1.319200	1.119800	2.552000	2.322200	2.544200	0.507000	3.478400	4.824400	8.889200	9.771600	0.097800	6.593000	3.580600	2.185200	6.134615	
pendigits	4.884800	10.256400	10.256400	10.256400	10.256400	10.256400	0.029600	0.026600	0.017200	0.054400	0.010800	3.531800	0.888800	0.979600	0.631600	0.626800	0.963600	0.425400	2.619800	2.477000	5.026400	4.988400	0.137200	4.448400	2.729600	2.319000	3.428854	
mnist	80.416000	36.290000	36.290000	36.290000	36.290000	36.290000	0.039400	0.020200	0.068200	0.251800	0.060000	74.250000	5.864400	5.912800	4.318200	4.665600	5.747100	0.442000	8.765200	15.727200	25.550000	28.662800	0.364200	31.341000	3.198400	2.920000	18.091415	
shuttle	6.191800	14.608800	14.608800	14.608800	14.608800	14.608800	0.053400	0.073000	0.029600	0.963400	0.025800	3.233800	1.329200	1.241600	1.586000	1.587000	1.589200	16.219600	25.645600	25.148800	27.112200	4.806200	5.177000	13.138000	8.778800	20.606800	17.843377	
satellite	8.561400	15.176800	15.176800	15.176800	15.176800	15.176800	0.069800	0.068800	0.029600	0.320200	0.015200	11.162800	0.975400	0.849400	0.842400	0.848400	0.886800	0.618000	2.828000	7.064400	10.156800	10.700600	0.135600	22.910800	2.774200	2.280000	6.153515	
satimage-2	4.828400	14.963200	14.963200	14.963200	14.963200	14.963200	0.092400	0.047000	0.029200	0.446600	0.028400	10.288400	0.914600	2.111800	2.222000	0.831200	2.141600	0.560800	2.402000	5.841400	9.702200	9.980600	0.114400	4.274000	11.528000	2.209600	5.557715	
wine	1.285000	9.146400	9.146400	9.146400	9.146400	9.146400	0.008400	0.013200	0.006000	0.115200	0.003000	0.074200	0.839000	0.816000	0.819200	0.814200	0.025400	0.125400	0.028400	0.005000	0.074000	0.072200	0.002000	1.589800	2.214200	1.792200	2.201400	
vowels	1.743400	10.425600	10.425600	10.425600	10.425600	10.425600	0.019400	0.016600	0.066800	0.083200	0.005000	8.966600	0.862200	0.840600	0.844200	0.828800	0.847800	0.083800	0.246600	0.065800	1.087400	1.013200	0.006800	6.187800	1.998400	2.577423		
glass	1.300600	9.412400	9.412400	9.412400	9.412400	9.412400	0.008000	0.009000	0.004600	0.042600	0.003000	0.069400	0.652800	0.626800	0.625400	0.484200	0.621400	0.006200	0.033200	0.003600	0.691600	0.080800	0.011600	1.956200	1.920600	1.497800	2.196592	
wbc	1.361000	12.587600	12.587600	12.587600	12.587600	12.587600	0.015000	0.012200	0.011800	0.118400	0.005000	0.523200	0.684400	0.694000	1.489800	1.471600	1.470200	1.182000	0.107800	0.018800	0.157000	0.109000	0.003000	1.715200	2.862300	1.904800	2.909662	
boston	1.469800	10.422200	10.422200	10.422200	10.422200	10.422200	0.013200	0.017000	0.006200	0.061800	0.003400	0.061800	0.572400	0.516400	0.518200	0.572000	0.620000	0.016600	0.085000	0.036200	1.143000	1.078000	0.003200	1.719000	3.928600	1.645200	2.565623	
WTN	0.000000	0.884615																										
AVG	21.483067	22.384876	22.384876	22.384876	22.384876	22.384876	0.148303	0.069971	0.230152	0.031771	8.987571	1.568319	3.312495	3.318181	3.301257	2.488981	1.669743	2.670790	14.091962	13.752019	12.261590	0.407943	6.377267	4.002648	3.184410	8.258909		
RCP	18.000000	21.095238	21.095238	21.095238	21.095238	21.095238	4.619048	4.285714	2.383333	6.285714	2.383333	11.904762	12.069528	11.714286	7.333333	11.389052	9.714286	16.571429	15.904762	4.380952	17.904762	17.380952	16.000000	16.000000	16.000000	16.000000	12.829760	
RK	21.000000	22.000000	22.000000	22.000000	22.000000	22.000000	3.000000	2.000000	6.000000	15.000000	11.000000	13.000000	12.000000	10.000000	7.000000	9.000000	8.000000	18.000000	16.000000	4.000000	4.000000	4.000000	4.000000	4.000000	4.000000	4.000000	17.000000	13.115385

CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD

45

Table A16: Model- and hyperparameters used for dataset specific parameter tuning.

param-alg	20news	arrhythmia	cardio	amuthyroid	breastw	letter	thyroid	manmography	pinna	mask	optdigits	pendigits	mnist	shuttle	satellite	svmimage-2	wine	vowels	glass	wbc	boston	
n_components-CMO	0.5	1.0	1.0	0.25	1.0	0.25	1.0	0.75	1.0	1.0	0.5	1.0	1.0	0.5	0.75	0.5	1.0	1.0	1.0	0.25	1.0	
n_components-CMO+	0.25	1.0	1.0	0.75	1.0	1.0	0.25	0.75	1.0	0.25	0.75	0.25	1.0	0.75	1.0	1.0	0.25	1.0	0.75	0.25	1.0	
n_components-CMO+k	0.25	0.25	0.25	0.75	1.0	1.0	0.25	0.75	1.0	0.25	0.5	0.25	1.0	1.0	1.0	1.0	0.25	1.0	0.5	0.25	1.0	
n_components-CMO+e	0.25	0.25	1.0	0.75	1.0	1.0	0.25	0.75	1.0	0.25	0.75	0.25	1.0	1.0	0.5	0.25	0.25	0.75	0.75	1.0	0.5	
n_components-CMO+ke	0.25	0.25	1.0	0.75	1.0	0.5	0.25	0.75	1.0	0.25	0.75	0.25	0.75	0.75	0.5	0.25	0.25	0.75	0.5	1.0	0.5	
n_components-CMOEms	1.0	1.0	1.0	0.75	1.0	1.0	0.75	0.75	0.25	0.5	0.75	1.0	1.0	0.75	0.5	1.0	0.75	1.0	0.5	1.0	0.75	
n_components-PCA	0.75	0.75	0.9999	0.25	0.75	0.25	0.25	0.75	0.9999	0.9999	0.9999	0.75	0.25	0.25	0.25	0.9999	0.25	0.75	0.25	0.5	0.9999	
n_components-PCA(NM)	0.25	0.75	1.0	1.0	0.25	1.0	1.0	1.0	1.0	0.25	0.5	1.0	1.0	0.25	1.0	1.0	1.0	1.0	1.0	1.0	0.25	
n_components-PCA(t)	0.25	1.0	0.25	0.75	0.25	0.75	1.0	0.5	0.25	0.75	0.25	0.5	1.0	0.25	0.25	0.25	1.0	1.0	0.75	0.25	0.25	
n_components-PCA(k)	1.0	0.75	1.0	1.0	0.25	0.75	1.0	0.5	1.0	0.25	0.25	0.75	1.0	0.25	0.25	1.0	1.0	1.0	0.75	0.25	0.25	
n_components-PCA(e)	1.0	0.75	0.25	0.75	0.5	0.75	1.0	0.5	0.25	0.75	1.0	0.25	0.75	0.75	0.25	0.5	1.0	1.0	0.75	1.0	0.25	
n_components-PCA(ke)	1.0	0.75	0.75	0.75	0.5	0.75	1.0	0.5	0.25	0.75	1.0	0.25	0.75	0.75	0.25	0.25	1.0	0.5	0.25	1.0	1.0	
n_components-PCA(Ems)	0.75	0.5	0.25	0.75	0.5	0.5	0.25	0.25	0.5	0.25	1.0	0.75	1.0	0.75	0.25	1.0	1.0	1.0	0.75	1.0	0.5	
n_components-PCA-MAD	384	274	6	5	9	32	5	3	4	166	16	8	75	7	36	18	13	12	7	8	11	
n_components-PCA-MAD++	192	274	11	6	7	32	3	2	8	166	16	4	50	9	36	18	4	12	3	8	4	
n_neighbors-LOF	5	95	100	100	100	10	100	100	100	5	20	10	100	100	100	100	100	55	10	95	75	
n_neighbors-KNN	5	45	100	15	100	5	45	45	80	100	100	100	65	100	100	100	100	5	5	90	55	
n_estimators-IF	10	50	90	10	40	10	10	60	20	100	100	10	50	90	70	100	30	20	10	30	70	
n_bins-HBOS	10	10	10	90	10	100	90	10	40	20	20	90	10	90	70	90	100	40	10	70	10	
m-OCSVM	0.1	0.7	0.9	0.1	0.9	0.5	0.1	0.2	0.9	0.4	0.1	0.4	0.9	0.9	0.2	0.8	0.3	0.2	0.1	0.6	0.6	
support_fraction-MCD	0.3	0.1	1.0	0.8	0.6	0.6	0.8	1.0	0.1	1.0	0.5	0.3	0.1	0.8	0.4	0.3	0.6	0.9	0.4	1.0	0.8	
support_fraction-EllipticEnv	1.0	1.0	1.0	0.2	0.6	0.6	1.0	1.0	0.1	1.0	1.0	0.3	0.3	1.0	0.4	0.3	0.6	0.9	0.4	1.0	0.8	
none-MLE	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
batch_size-AE	8	256	16	32	8	4	8	256	64	256	16	32	4	64	4	32	16	4	4	32	32	
batch_size-VAE	64	256	256	256	4	128	256	16	8	256	64	256	128	256	8	8	4	4	4	32	8	4
batch_size-DeepSVDD	16	64	256	64	8	32	256	256	4	32	256	256	256	32	256	256	4	64	64	16	128	

References

- Aggarwal, C.C. (2016). *Outlier analysis* (2nd ed.). New York: Springer.
- An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1), 1–18.
- Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data* (p. 93–104). New York, NY, USA: Association for Computing Machinery.
- Campos, G.O., Zimek, A., Sander, J., Campello, R.J., Micenková, B., Schubert, E., . . . Houle, M.E. (2016, jul). On the Evaluation of Unsupervised Outlier Detection: Measures, Datasets, and an Empirical Study. *Data Min. Knowl. Discov.*, 30(4), 891–927.
- Candès, E.J., Li, X., Ma, Y., Wright, J. (2011). Robust principal component analysis? *J. ACM*, 58(3), 11.
- Crammer, K., & Chechik, G. (2004). A Needle in a Haystack: Local One-Class Optimization. *Proceedings of the Twenty-First International Conference on Machine Learning* (p. 26).
- Davis, J., & Goadrich, M. (2006). The Relationship between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine Learning* (p. 233–240).
- Depersin, J., & Lecué, G. (2021). On the robustness to adversarial corruption and to heavy-tailed data of the Stahel-Donoho median of means. *preprint arXiv:2101.09117*.
- Falk, M. (1997). On Mad and Comedians. *Annals of the Institute of Statistical Mathematics*, 49(4), 615–644.
- Goldstein, M., & Dengel, A. (2012). Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 9.
- Han, S., Hu, X., Huang, H., Jiang, M., Zhao, Y. (2022). Adbench: Anomaly detection benchmark. S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (Vol. 35, pp. 32142–32159). Curran Associates, Inc.
- Hinton, G.E., & Salakhutdinov, R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*, 313, 504 - 507.
- Huang, Y., Jin, W., Yu, Z., Li, B. (2021). A robust anomaly detection algorithm based on principal component analysis. *Intelligent Data Analysis*, 25(2), 249–263.

- Hubert, M., Rousseeuw, P.J., Verdonck, T. (2010). *A DETERMINISTIC ALGORITHM FOR THE MCD* (Tech. Rep.). Celestijnenlaan 200B, B-3001 Leuven (Heverlee): Section of Statistics, Department of Mathematics, Katholieke Universiteit Leuven, The Netherlands.
- Jolliffe, I. (1986). *Principal Component Analysis*. New York: Springer.
- Kazempour, D., Hünemörder, M.A.X., Seidl, T. (2019). On comads and principal component analysis. *Similarity Search and Applications - 12th International Conference, SISAP 2019, Newark, NJ, USA, Proceedings* (pp. 273–280).
- Kriegel, H.-P., Kröger, P., Schubert, E., Zimek, A. (2008). A General Framework for Increasing the Robustness of PCA-Based Correlation Clustering Algorithms. *Proceedings of the 20th International Conference on Scientific and Statistical Database Management* (p. 418–435).
- Liu, F.T., Ting, K.M., Zhou, Z.-H. (2012). Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data*, 6(1).
- Maronna, R.A., & Yohai, V.J. (1995). The Behavior of the Stahel-Donoho Robust Multivariate Estimator. *Journal of the American Statistical Association*, 90(429), 330-341.
- Moors, J.J.A. (1986). The meaning of kurtosis: Darlington reexamined. *The American Statistician*, 40(4), 283–284.
- Peña, D., & Prieto, F.J. (2001). Multivariate outlier detection and robust covariance matrix estimation. *Technometrics*, 43, 286 - 310.
- Rousseeuw, P.J. (1984). Least median of squares regression. *Journal of the American Statistical Association*, 79(388), 871–880.
- Rousseeuw, P.J., & Driessen, K.V. (1999). A Fast Algorithm for the Minimum Covariance Determinant Estimator. *Technometrics*, 41(3), 212–223.
- Ruff et al. (2018, 10–15 Jul). Deep One-Class Classification. *Proceedings of the 35th International Conference on Machine Learning* (Vol. 80, pp. 4393–4402). PMLR.
- Sajesh, T.A., & Srinivasan, M.R. (2012). Outlier detection for high dimensional data using the Comedian approach. *Journal of Statistical Computation and Simulation*, 82(5), 745-757.
- Shyu, M., Chen, S., Sarinnapakorn, K., Chang, L. (2003). A novel anomaly detection scheme based on principal component classifier. *in Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM'03)* (pp. 172–179).

48 References

- Zhao, Y., Rossi, R., Akoglu, L. (2021). Automatic Unsupervised Outlier Model Selection. M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J.W. Vaughan (Eds.), *Advances in Neural Information Processing Systems* (Vol. 34, pp. 4489–4502). Curran Associates, Inc.

Chapter 4

AnyCORE: An Anytime Algorithm for Cluster Outlier REmoval

This chapter consists of a preprint version of the following publication:

Andreas Lohrer, Anna Beer, Maximilian Archimedes Xaver Hünemörder, Jenny Lauterbach, Thomas Seidl, and Peer Kröger. “AnyCORE - An Anytime Algorithm for Cluster Outlier REmoval”. In: *Proceedings of the LWDA 2021 Workshops: FGWM, KDML, FGWI-BIA, and FGIR, Online, September 1-3, 2021*. Ed. by Thomas Seidl, Michael Fromm, and Sandra Obermeier. Vol. 2993. CEUR Workshop Proceedings. CEUR-WS.org, 2021, pp. 145–156. URL: <https://ceur-ws.org/Vol-2993/paper-14.pdf>

- **Conception:** Beer (Lead), **Lohrer** (Support)
- **Planning:** Beer (Lead)
- **Execution:** Lauterbach (Lead), **Lohrer** (Support)
- **Manuscript:** Beer (Lead), **Lohrer** (Support), Hünemörder (Support), Kröger (Support), Seidl (Support)

AnyCORE - An Anytime Algorithm for Cluster Outlier REmoval

Andreas Lohrer¹, Anna Beer², Maximilian Hünemörder¹, Jennifer Lauterbach²,
Thomas Seidl² and Peer Kröger¹

¹Christian-Albrechts-University Kiel, Germany, Christian-Albrechts-Platz 4, 24118 Kiel, Germany

²Ludwig-Maximilians-Universität München, Germany, Oettingenstraße 67, 80538 München, Germany

Abstract

We introduce AnyCORE (Anytime Cluster Outlier REmoval), an algorithm that enables users to detect and remove outliers at anytime. The algorithm is based on the idea of MORE++[1], an approach for outlier detection and removal that iteratively scores and removes $1d$ -cluster-outliers in n -dimensional data sets. In contrast to MORE++, AnyCORE provides continuous responses for its users and converges independent of cluster centers. This allows AnyCORE to perform outlier detection in combination with an arbitrary clustering method that is most suitable for a given data set. We conducted our AnyCORE experiments on synthetic and real-world data sets by benchmarking its variant with k -Means as the underlying clustering method versus the traditional batch algorithm version of MORE++. In extensive experiments we show that AnyCORE is able to compete with the related batch algorithm version.

Keywords

anytime algorithm, outlier removal, outlier detection, clustering

1. Introduction

In many domains, e.g., in sports, health, or mobility, data-driven applications and the connected business processes benefit from algorithms capable of detecting and removing outliers at any time. However, since most outlier detection algorithms work in batch mode¹ and the amounts of most commonly massive and high-dimensional data sets are likely to be computationally expensive, applications and users face long response times, avoiding to take fast decisions. Furthermore, outliers in the data set can have a distortive impact on the accuracy of results and cause additional unnecessary computational efforts. As digitalization is essential for various domains and business areas, an increase in the amount of structured and unstructured data generated by related applications is the typical consequence. Leveraging these amounts of data means finding insightful patterns which can be used for subsequent information reasoning or to derive recommendations for further actions. Such patterns can be, e.g., clusters of similar data points or anomalous data points, which we call outliers. Finding clusters and identifying

LWDA'21: Lernen, Wissen, Daten, Analysen September 01–03, 2021, Munich, Germany

✉ alo@informatik.uni-kiel.de (A. Lohrer); beer@dbs.ifi.lmu.de (A. Beer); mah@informatik.uni-kiel.de (M. Hünemörder); jenny.lauterbach@fs.lmu.de (J. Lauterbach); seidl@dbs.ifi.lmu.de (T. Seidl); pkr@informatik.uni-kiel.de (P. Kröger)

🆔 0000-0001-7834-301X (A. Lohrer); 0000-0002-4861-1412 (T. Seidl); 0000-0001-5646-3299 (P. Kröger)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).



CEUR Workshop Proceedings (CEUR-WS.org)

¹returning results only if the computation has been completely finished

outliers are tasks that become increasingly cumbersome, in terms of runtime, with an increasing number of observations and feature dimensions. In addition to that, noise observations that are neither part of any cluster nor a significant outlier are not only irrelevant for data analysts[2], they can also have negative effects on the results of clustering algorithms. Thus, excluding them can optimize the resulting cluster structures. Since clustering and outlier removal are already challenging tasks for low-dimensional data, batch mode¹ algorithms cannot provide the related responses at any time to its users for nowadays often large or high-dimensional data.

Thus, we introduce in this paper AnyCORE, an **Any**time Cluster **O**utlier **R**emoval algorithm, which allows users to request intermediate results at any time, even before the algorithm has finished. Optimizing the algorithm MORE++[1] our algorithm not only alternates between the originally consecutive clustering and outlier removal steps of MORE++, it also provides intermediate results at any time while allowing the user to select clustering and outlier decision methods most suitable according to the provided data set.

The main contributions of this paper are:

- The k -Means Outlier Removal algorithm gets enhanced to provide users response with intermediate results at any time during its runtime.
- AnyCORE utilizes removed outliers to get more precise clustering results by alternating between clustering and outlier removal.
- AnyCORE allows users to interrupt, to check the results, and to stop the algorithm at any time if the results are considered as satisfactory.
- The user can select the underlying clustering and outlier scoring method suitable to the analyzed data set.

The remaining paper is structured as follows. Section 2 describes related work for anytime algorithms, clustering and outlier removal algorithms. In Section 3, we describe our algorithm AnyCORE and evaluate its competitiveness in Section 4. Section 5 concludes the paper.

2. Related Work

In this section, we provide an overview of the related literature. This includes the description of related anytime clustering and outlier detection algorithms, as well as the introduction of the underlying outlier removal algorithm MORE++[1] which got optimized by our AnyCORE approach.

MORE++. MORE++[1] is a k -Means++ based algorithm for **O**utlier **R**emoval which is also capable to perform on high-dimensional data. It performs k -Means++[3] until convergence to unify similar observations within clusters. For the subsequent step of outlier detection the algorithm analyzes the given samples in each single dimension by creating 1d-projections separately for each single cluster (see Figure 1). The resulting histograms of each cluster in each single dimension allow even visually the identification of outliers by gaps or significant outlying rises or falls among its frequencies. Counting the single dimensions in which a sample got detected as cluster outlier allows one to calculate an overall outlier score for a sample and

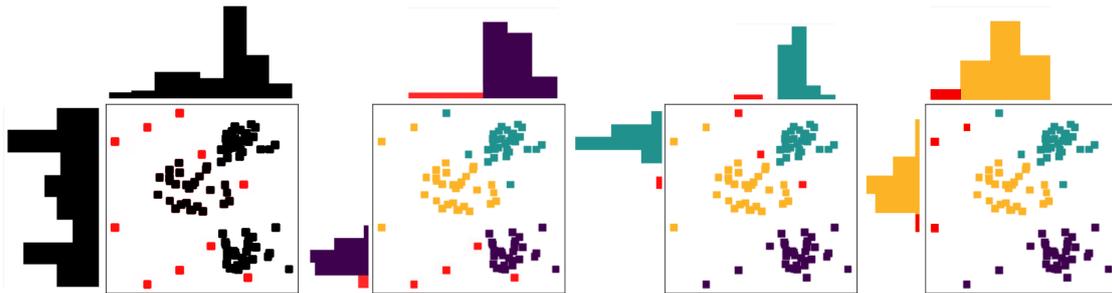


Figure 1: This illustration visualizes outliers (red) detected by MORE++ [1] during its clusterwise 1-dimensional outlier analysis. The black histogram on the left shows the aggregated result of the single cluster-based outlier analysis on the right. (based on Fig. 1.[1])

distinguish normal from abnormal samples by a given threshold. This enables the algorithm to flag samples as outliers and additionally for their removal.

Since MORE++ does not consider the beneficial effect of outlier removals upfront of its clustering step we address this potential within our AnyCORE approach in Section 3.

***k*-means**— The concept of combining clustering and outlier detection by removing outliers before the clustering step was also introduced by, e.g., *k*-Means—[4]. The authors provide an extension to *k*-Means, where a fixed number l of points are removed in each iteration, before the centroids are computed. This is achieved by ranking the points in the input data set X by distance from the current centroids. Then the next centroids are computed from the set difference of X and the l points furthest away from the centroids. For the next iteration the furthest points are recomputed and again removed from the original data set X . At convergence the algorithm results in a set of cluster centers and a set of exactly l outliers. Note that *k*-Means— uses the removal of the furthest points in order to optimize clustering results of *k*-Means, where our algorithm AnyCORE is designed to find outliers in high-dimensional space, working independently of the cluster algorithm. AnyCORE does not need necessarily any cluster centroids or means as provided by *k*-Means. Due to the curse of dimensionality, the Euclidean distance used by *k*-Means— can not be used to discern distances to the cluster centers for very high dimensionalities any more, where our histogram-based solution is suitable also for large numbers of attributes in the data set.

Anytime algorithms. The concept of anytime algorithms has already been introduced in 1988 by Dean & Boddy [5, 6] which follows the approach that an algorithm can be stopped at anytime during computation in order to quickly provide its users intermediate results with a reasonable result quality. This allows the users to decide if the given results are satisfactory or if the processing of the algorithm should be resumed in order to receive a further improved result. The quality of early results should be close to the quality of a batch algorithm and the final result should be at least as good or even better than the batch algorithm. The runtime of an anytime algorithm should not be much larger than the batch algorithm's runtime [7, 8, 9]. This behavior is illustrated in Figure 2.

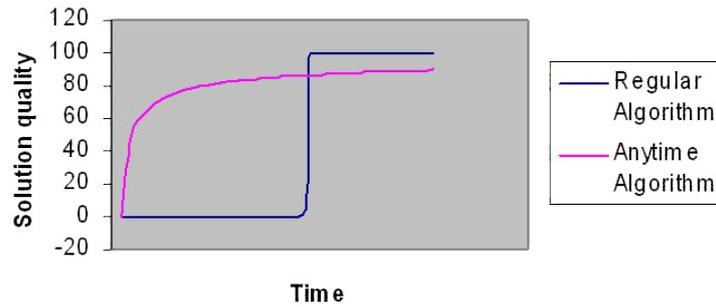


Figure 2: Comparison of the result quality in relation to the runtime for an anytime and batch algorithm.

Since this approach of approximated processing [10] has been proven to be useful for AI applications [7] it has been evolved and widely adapted to various tasks in the field of knowledge discovery. This led to developments like, e.g., AnyDBC [11, 12], which is a DBSCAN-based anytime clustering algorithm that can be used on high-dimensional data sets like time series or trajectories. Further developments are AnyHC [13], an anytime algorithm for hierarchical clustering, and different variants of A-DBSCAN [14, 15] which deliver results based on approximated density-based clusters. I-kMeans [16] works with Discrete and Haar Wavelet Transformation as an anytime k -Means on time series and was extended with Piecewise Constant Approximation (PAA) in [17]. These algorithms are just a few well known representants of anytime algorithms, which mainly perform clustering on data streams. Besides the tasks of clustering and anytime embedding [18], anytime outlier detection has also been addressed with algorithms like, e.g., AnyOut [19], which uses the data structure ClusTree [20] to analyze constant and changing data streams. In addition to that, there are approaches like FPOF [21] for frequent pattern outlier detection on massive data sets or the anytime novelty detection method of Sofman [22].

This selection of anytime algorithms gets extended by our proposed method AnyCORE which is capable to conduct clustering as well as outlier detection and removal within one iteration so that its users are constantly able to stop, interrupt and request intermediate results at any time.

3. AnyCORE

In this section we introduce our algorithm AnyCORE (**A**nymtime **C**luster **O**utlier **R**emoval) and highlight its differences to the underlying algorithm MORE++[1] in detail. Variables and methods from the pseudo code of Algorithm 1 are referenced and formatted in *italics*.

Revisiting the MORE++ approach, which we described in detail in Section 2, we observe that the removal of outliers holds further potential for optimization. This potential gets revealed by our method utilizing the beneficial effects of removed outliers in the clustering phase, which is without outliers more likely to calculate correct cluster labels, as, e.g., already applied in k -Means— introduced in Section 2. This gets achieved by alternating between clustering and

outlier removal instead of running them consecutively. In addition to that an outlier score threshold *ost* is required. The threshold is needed, because AnyCORE temporarily removes outliers from the data set to get a more precise result. Since the convergence criteria of the percentage of cluster jumpers² is independent of the convergence and availability of cluster centers, AnyCORE is able to run with arbitrary clustering methods and arbitrary outlier scoring methods, which allows to choose those methods which are most suitable for the provided data set. Hence the last input parameters of AnyCORE are the selected clustering algorithm *algo* with its related parameter set *p* and the selected outlier scoring method *outlier_scoring*. The optional parameter *init* can be used to initialize centers in case of *algo* is centroid-based but also to e.g. provide an initial ϵ -range, which can be adapted according to *1d*-gap-sizes in case of *algo* is density-based. For our experiments in Section 4 we evaluated our method with the clustering algorithms *k*-Means++ (*p=k*, *init='k-means++'*) as well as with the histogram-based outlier-scoring method of MORE++ which considers the ratio of outlier dimensions of a sample as outlier score (parameter *outlier_scoring*). AnyCORE can be interrupted by users at any time to get an intermediate result. If users are not satisfied with the outcome, they can either stop the algorithm and start again with other input parameters or continue the current calculation to get a refined result. For the calculation of intermediate results, the *outlierScore* or the *numberOfOutlierDims* is needed as it changes the outcome of the *outlierScore*. Thus the result of AnyCORE can change whenever *numberOfOutlierDims* is updated after each calculation of the *1dOutliers*. Only if users stop AnyCORE during the first iteration, an intermediate *outlierScore* with the newest value of *numberOfOutlierDims* gets returned without previously removed outliers. As in each iteration outliers and the *numberOfOutlierDims* are newly calculated and as up from the second iteration an refined initialization by outlier removal is applied, one faces potentially also partly incorrect cluster memberships. So it can be possible for interim results to get intermediately worse than previous interim results. The pseudo-code of the illustrated Algorithm 1 describes the sequences of AnyCORE. Lines 9-27 represent the part of MORE++ but with the above described AnyCORE-specific adaptations highlighted by the prefix symbol #.

4. Experiments

In this section we describe our experimental setup as well as the results of our experiments demonstrating the competitiveness of our method AnyCORE.

Datasets. The experiments for benchmarking the AnyCORE approach have been conducted on synthetic as well as on real world data sets. Since massive real world data sets are scarcely available for outlier detection, all used ODDS³ data sets described in the Table 1 had been evaluated despite of their short runtime. Additionally to those data sets, we also experimented with two constructed synthetic data sets, SynthA and SynthB, in order to investigate how AnyCORE performs on large data sets. Both synthetic data sets had been constructed using cluster centers drawn from a uniform distribution function and generating gaussian-distributed clusters around them (parameters: *centers=5*, *center_box=(0, 40)*, *cluster_std=1.2*, *random_state=3*). In

²samples which changed their cluster membership between the previous and current iteration of AnyCORE

³<http://odds.cs.stonybrook.edu/>

Algorithm 1: Pseudo-Code of AnyCORE

Data: Data X , #clustering algorithm $algo$, clustering algorithm parameters $p \leftarrow \{\text{number of clusters } k\}$, outlier score threshold ost , #scoring method $outlier_scoring$, clustering initialization $init \leftarrow \{\text{'k-means++'}\}$

Result: Clustering labels, $outlierScore$ for data points

```

1  $t\_start \leftarrow now$ ;
2  $outliers \leftarrow$  empty list;
3  $clusterMembershipsOld \leftarrow$  empty list;
4 while not converged and not stopped do
5   if  $outliers \neq$  empty list then
6      $clusterMemberships, init \leftarrow \#algo(X \setminus outliers, p, init)$ ;
7   end
8    $clusterMemberships, init \leftarrow \#algo(X, p, init)$ ;
9   foreach  $x \in X$  do
10     $numberOfOutlierDims[x] \leftarrow 0$ ;
11  end
12  foreach  $c \in clusters$  do
13    foreach  $d \in range(dimensions)$  do
14       $histogram \leftarrow buildHistogram(c, d)$ ;
15       $1dOutliers \leftarrow calculate1dOutliers(histogram)$ ;
16      foreach  $1dOutlier \in 1dOutliers$  do
17         $numberOfOutlierDims[1dOutlier] ++$ ;
18      end
19    end
20  end
21   $outliers \leftarrow$  empty list;
22  foreach  $x \in X$  do
23     $outlierScore[x] \leftarrow \#outlier\_scoring(numberOfOutlierDims[x])$ ;
24    if  $outlierScore[x] \geq ost$  then
25       $outliers.add(x)$ ;
26    end
27  end
28   $timestamp \leftarrow (now - t\_start)$ ;
29  publish  $timestamp, outlierScore$ ;
30  while interrupted do
31    sleep
32  end
33   $converged = (perc\_changed(clusterMemberships, clusterMembershipsOld) == 0)$ ;
34   $clusterMembershipsOld \leftarrow clusterMemberships$ ;
35 end

```

order to be able to properly benchmark the outlier detection and removal part of the compared algorithms, these data sets had been additionally extended by outliers, which also follow an uniform distribution function (parameters: low=0, high=40, seed=5).

Dataset	n	dim	outlier	k	type
Glass	214	9	9 (4.20%)	5	real world
Lympho	148	18	6 (4.10%)	2	real world
Mnist	7603	100	700 (9.20%)	2	real world
Musk	3062	166	97 (3.20%)	3	real world
Optdigits	5216	64	150 (3.00%)	9	real world
Shuttle	49097	9	3511 (7.00%)	1	real world
Smtpt	95156	3	30 (0.03%)	1	real world
SynthA	100000	5	5000 (5.00%)	5	synthetic
SynthB	1000	3000	50 (5.00%)	5	synthetic

Table 1: ODDS³ real world data sets and synthetic data sets used within the experiments.

Compared methods and experimental setup. We compare the performance of the regular batch algorithm MORE++ (cf. Section 2) with its adapted anytime version AnyCORE as introduced in Section 3. For our experiments we initialized the input parameters of the compared algorithms with the empirically determined values described in Table 2. The *outlier_scoring* method is implemented by the relative outlier dimension count ratio and the outlier scoring threshold *ost*, which decides if a sample is considered as an actual outlier in the full-dimensional space. Moreover, for the arbitrarily selectable clustering method *cluster_algo* of AnyCORE we decided to conduct the experiments with *k*-Means++ to compute clusters for the outlier detection phase but also to emphasize the beneficial effects generated by the anytime related adaptations like iterative outlier removal and anytime response. Furthermore, we used the Area Under the Receiver Operating Characteristic Curve (ROC AUC)⁴ score as evaluation metric for our experiments. Based on rates, ROC inherently adjusts for the imbalance of class sizes typical of outlier detection tasks.[23]

All experiments ran 100 times for each MORE++ due to *k*-Means++'s non-determinism. So it is possible that for some timestamps only a few results were measured, depending on how much time AnyCORE has needed to finish its computation. We took the average of these results for MORE++ and compared them to the average result of AnyCORE with respect to the respective timestamp. Whenever the intermediate result might change, we saved the current timestamp and the current *outlierCountDim* and calculated the outlier score for each timestamp after the computation is finished. Due to saving timestamps and scores for anytime response, AnyCORE needs partly more time than MORE++ (cf. Figure 3a, 3c and 3g).

⁴https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html (18.07.2021)

	<i>outlier_scoring</i> <i>ost</i>	<i>cluster_algo</i> <i>k-Means</i>
Glass	0.1	5
Lympho	0.4	2
Mnist	0.2	2
Musk	0.3	3
Optdigits	0.2	9
Shuttle	0.3	1
Smtp	0.4	1
SynthA	0.2	5
SynthB	0.2	5

Table 2: Empirically determined hyperparameters.

Result discussion. In this section we discuss the results of our experiments. We compared the runtime and the quality of the outlier detection over time with the batch algorithm MORE++ and our adapted anytime version AnyCORE. As further described in the experimental setup the outlier detection quality is represented by the ROC AUC evaluation metric, which gets illustrated in Figure 3 and Figure 4. Results better than randomness are indicated by ROC AUC Scores above 0.5.

The experiments showed that the result quality strongly depends on the data set. Observing the result plots in Figure 3 and Figure 4 one recognizes that AnyCORE gets better the more time it takes for computation. For the shuttle dataset in Figure 3d, our approach is even significantly faster than MORE++. Lympho in Figure 3b, MNIST in Figure 3g and Musk in Figure 3a have adequate results. The intermediate results are similar to the outcome of MORE++.

The plots of Glass in Figure 3f and Optdigits in Figure 3e are especially interesting, since in those AnyCORE has partly much better intermediate results than MORE++. But these two data sets also support the assumption from section 3 of potentially facing performance fluctuations based on the alternating outlier removals. Thus more experiments and evaluations about how the outliers behave and change during the iterations of AnyCORE are required. The experiments on the synthetic data sets in Figure 3h, 4a and 4b just show that saving *timestamp* and *outlierCountDim* takes especially much time for many dimensions. AnyCORE is comparably slow due to saving *timestamp* and *outlierScore*, but it provides intermediate results. Thus, for massive complex data sets AnyCORE is a reasonable choice, if users are looking for an anytime algorithm. Its comparison to MORE++ is more difficult, as MORE++ is often finished before even the first intermediate results got provided, but AnyCORE is likely to deliver more refined results with increasing runtime. Thus, before making a final decision, there should be evaluations conducted for each individual data set in order to investigate how and when AnyCORE provides better results.

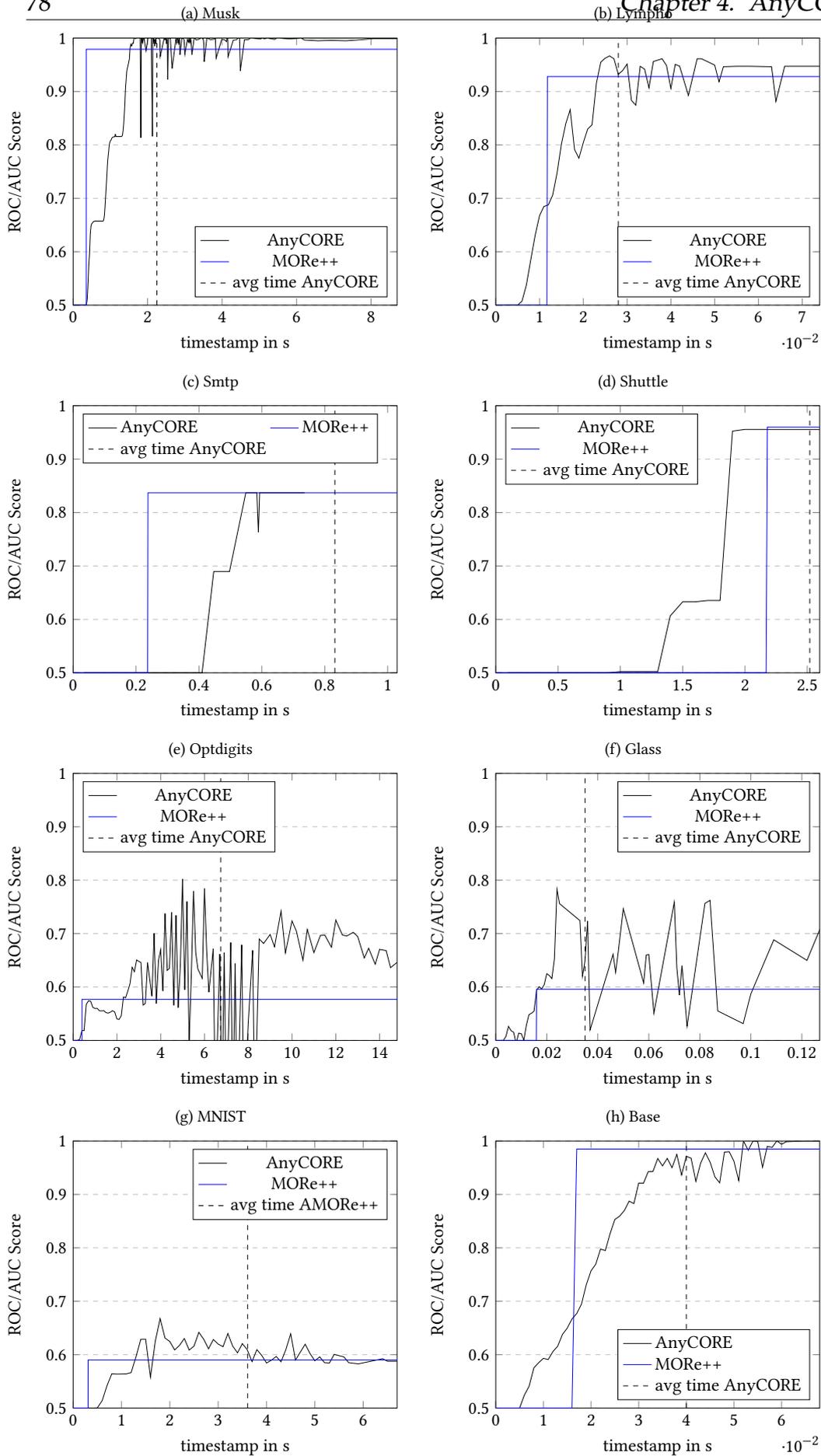


Figure 3: AnyCORE experiment results on base and real-world datasets.

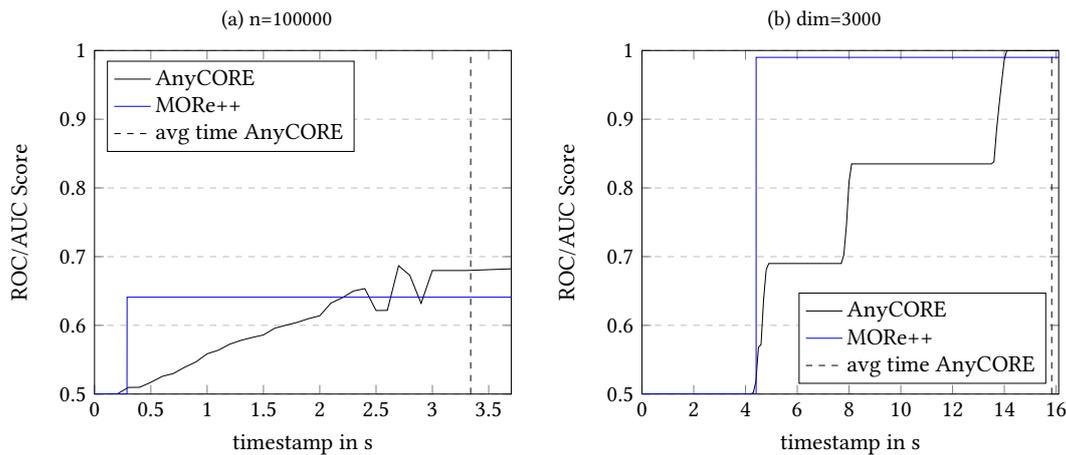


Figure 4: AnyCORE experiment results on synthetic datasets.

5. Conclusion

In summary, we introduced the Anytime Cluster Outlier Removal algorithm AnyCORE, which utilizes MORE++’s outlier detection approach relying on histograms of 1d-projections of separately regarded clusters. Profiting from MORE++’s advantages of not depending on distance measures to work in high-dimensional spaces and of non-exponential runtimes, AnyCORE enhances this outlier removal algorithm by allowing users to interrupt and check intermediate results at any time and to stop in case of reviewed results are considered satisfactory. The experiments demonstrated, that dependent on the individual data set, our anytime version tends to outperform its batch algorithm version MORE++. Since the AnyCORE convergence criteria of the percentage of cluster jumpers² is independent of the underlying clustering algorithm, further evaluations with alternative clustering methods than k -Means++ can be done in future work. Furthermore, the partly fluctuating intermediate results could be addressed by a moving average of the initialization values for the clustering algorithm. In addition to that, the current implementation of the histogram-based *outlier_scoring* method could be replaced and evaluated with alternative approaches like Kernel Density Estimation, which does not depend on the number of bins. Moreover, the runtime of MORE++ as well as of the proposed method AnyCORE would benefit furthermore from parallelization of the outlier detection and removal steps done for each single dimension.

Acknowledgments

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

References

- [1] A. Beer, J. Lauterbach, T. Seidl, More++: k-means based outlier removal on high-dimensional data, in: G. Amato, C. Gennaro, V. Oria, M. Radovanović (Eds.), *Similarity Search and Applications*, Springer International Publishing, Cham, 2019, pp. 188–202.
- [2] C. Aggarwal, *Outlier analysis*, in: Springer New York, 2013.
- [3] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: *SODA '07*, 2007.
- [4] S. Chawla, A. Gionis, k-means-: A unified approach to clustering and outlier detection, in: *13th SIAM International Conference on Data Mining*, Austin, Texas, 2013, 2013, pp. 189–197. VK: hiit.
- [5] M. Boddy, T. L. Dean, Deliberation scheduling for problem solving in time-constrained environments, *Artificial Intelligence* 67 (1994) 245–285.
- [6] T. Dean, M. Boddy, An analysis of time-dependent planning, in: *Proceedings of the Seventh AAAI National Conference on Artificial Intelligence*, AAAI'88, AAAI Press, 1988, p. 49–54.
- [7] S. Zilberstein, Using anytime algorithms in intelligent systems, *AI Mag.* 17 (1996) 73–83.
- [8] S. Zilberstein, S. J. Russell, *Approximate reasoning using anytime algorithms*, 1995.
- [9] J. Grass, S. Zilberstein, Anytime algorithm development tools, *SIGART Bull.* 7 (1996) 20–27.
- [10] V. R. Lesser, J. Pavlin, E. Durfee, Approximate processing in real-time problem solving, *AI magazine* 9 (1988) 49–49.
- [11] S. T. Mai, I. Assent, M. Storgaard, Anydbc: An efficient anytime density-based clustering algorithm for very large complex datasets, in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, Association for Computing Machinery, 2016, p. 1025–1034.
- [12] S. T. Mai, I. Assent, J. Jacobsen, M. S. Dieu, Anytime parallel density-based clustering 32 (2018) 1121–1176.
- [13] O. Arslan, D. E. Koditschek, Anytime hierarchical clustering, 2014. [arXiv:1404.3439](https://arxiv.org/abs/1404.3439).
- [14] C. Böhm, J. Feng, X. He, S. Mai, Efficient anytime density-based clustering, in: *SDM*, 2013.
- [15] S. Mai, X. He, J. Feng, C. Plant, C. Böhm, Anytime density-based clustering of complex data, *Knowledge and Information Systems* 45 (2014) 319–355.
- [16] J. Lin, M. Vlachos, E. J. Keogh, D. Gunopulos, Iterative incremental clustering of time series, in: *EDBT*, 2004.
- [17] J. Lin, M. Vlachos, E. J. Keogh, D. Gunopulos, J. Liu, S. Yu, J. jin Le, A mpaa-based iterative clustering algorithm augmented by nearest neighbors search for time-series data streams, in: *PAKDD*, 2005.
- [18] A. Tsitsulin, M. Munkhoeva, D. Mottin, P. Karras, I. Oseledets, E. Müller, Frede: Anytime graph embeddings, *Proc. VLDB Endow.* 14 (2021) 1102–1110.
- [19] I. Assent, P. Kranen, C. Baldauf, T. Seidl, Anyout: Anytime outlier detection on streaming data, in: *DASFAA*, 2012.
- [20] P. Kranen, I. Assent, C. Baldauf, T. Seidl, The clustree: indexing micro-clusters for anytime stream mining, *Knowledge and Information Systems* 29 (2010) 249–272.
- [21] A. Giacometti, A. Soulet, Anytime algorithm for frequent pattern outlier detection, *Inter-*

- national Journal of Data Science and Analytics 2 (2016) 119–130.
- [22] B. Sofman, J. Andrew Bagnell, A. Stentz, Anytime online novelty detection for vehicle safeguarding, in: 2010 IEEE International Conference on Robotics and Automation, 2010, pp. 1247–1254. doi:10.1109/ROBOT.2010.5509357.
- [23] A. Zimek, On the evaluation of unsupervised outlier detection, 2016. URL: <https://imada.sdu.dk/~zimek/InvitedTalks/TUVienna-2016-05-18-outlier-evaluation.pdf>, talk at TU Vienna (last access: 18.07.2021).

Chapter 5

A Transparent Transformer Model for Group Anomaly Detection on Trajectories

This chapter consists of a preprint version of the following publication:

Andreas Lohrer, Darpan Malik, Claudius Zelenka, and Peer Kröger. “GAD-former: A Transparent Transformer Model for Group Anomaly Detection on Trajectories”. In: (2024). arXiv: 2303.09841 [cs.LG]

- **Conception:** Lohrer (Lead)
- **Planning:** Lohrer (Lead)
- **Execution:** Lohrer (Lead), Malik (Support)
- **Manuscript:** Lohrer (Lead), Malik (Support), Zelenka (Support), Kröger (Support)

GADformer: A Transparent Transformer Model for Group Anomaly Detection on Trajectories

Andreas Lohrer, Darpan Malik, Claudius Zelenka, Peer Kröger
Information Systems and Data Mining, Kiel University
 Kiel, Germany
 {alo,cze,prk}@informatik.uni-kiel.de, stu225397@mail.uni-kiel.de

Abstract—Group Anomaly Detection (GAD) identifies unusual pattern in groups where individual members might not be anomalous. This task is of major importance across multiple disciplines, in which also sequences like trajectories can be considered as a group. As groups become more diverse in heterogeneity and size, detecting group anomalies becomes challenging, especially without supervision. Though Recurrent Neural Networks are well established deep sequence models, their performance can decrease with increasing sequence lengths. Hence, this paper introduces GADformer, a BERT-based model for attention-driven GAD on trajectories in unsupervised and semi-supervised settings. We demonstrate how group anomalies can be detected by attention-based GAD. We also introduce the Block-Attention-anomaly-Score (BAS) to enhance model transparency by scoring attention patterns. In addition to that, synthetic trajectory generation allows various ablation studies. In extensive experiments we investigate our approach versus related works in their robustness for trajectory noise and novelties on synthetic data and three real world datasets.

Index Terms—Group Anomaly Detection, BERT, Model Inspection, Trajectories, Deep Learning, Artificial Intelligence

I. INTRODUCTION

Group Anomaly Detection (GAD) is an important task across many disciplines and domains like computational fluid dynamics and computer vision [1, 2], mobility [3, 4], physics [5, 6], social networks [7] and many more. In these domains, GAD is suitable for various types of group anomalies. Since group member instances can be an arbitrary representation, the GAD paradigm also applies to the detection of anomalous sequences like trajectories, to which [8] refers to as collective anomalies. Especially in the spatio-temporal domain, Trajectory Anomaly Detection is a common task to reveal abnormal behavior as the authors [9, 10, 3] confirm.

However, although sequential coordinates of a trajectory obviously represent a group structure, the detection of individual anomalous trajectories has not been addressed as a group anomaly detection problem yet and not at all with transparent multi-head attention.

The current state of the art approaches for anomaly detection on trajectories are recurrent neural networks (RNNs) like LSTMs [11] and GRUs [12, 10], but the potential of deep learning methods for Group Anomaly Detection has rather been sparsely investigated. So far GAD tasks have more likely been solved by generative topic models [13, 7, 1] or SVM-based methods [14, 5, 4]. Despite recent advances,

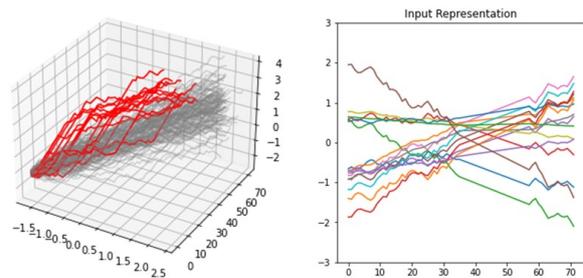


Fig. 1. GADformer trajectory representations on synthetic trajectory data - on the left: 72 raw trajectory steps p_n (=group members o_n) as part of gray normal or red abnormal groups (=individual trajectories Λ_m); on the right: trajectory step embeddings e_n of one individual trajectory (=group \mathcal{G}).

deep generative models got only involved in form of Adversarial Autoencoders (AAEs) and Variational Autoencoders (VAEs) [2] to perform GAD for images. [3] offers different machine learning based algorithms to detect anomalous groups of multiple trajectories, but does not identify the detection of individual anomalous trajectories (a sequence of group members) as group anomaly detection problem.

However, anomalous behavior is not ensured to just appear within short trajectory segments. It is challenging for recurrent neural networks, sometimes even LSTM and GRU, to learn very long-term dependencies [10].

A further challenge for deep learning based group anomaly detection on trajectories is, that although trajectory data is highly available, it is rather weakly labeled or does not overcome the nonground-truth problem [9] at all.

In order to tackle these challenges we introduce our approach GADFormer, a BERT[15] based architecture with transformer[16] encoder blocks for attention-based group anomaly detection on trajectories. Extending the idea of [16] to optimize also image, audio or video sequence tasks by their transformer approach, we identify transformer based models for a sequence of trajectory points/segments as group member instances of a group anomaly detection task as similarly beneficial. Our model can be trained in an unsupervised as well as in a semi-supervised setting so that there is no or only reduced need for labeled trajectories. Furthermore, we introduce a Block Attention-anomaly Score (BAS), which allows us to provide an transparent view to the capability of the transformer

encoder blocks to distinguish normal from abnormal trajectory attention matrices. We show with extensive experiments on synthetic and real world datasets that our approach is on par with the state of the art methods like GRU or MainTulGAD, an adapted version of [17] for GAD.

Hence, the contributions of our work can be summarized as follows:

- Transformer-Encoder-architecture capable to perform attention-based group anomaly detection in an unsupervised and semi-supervised setting.
- Identification of the detection of individual anomalous trajectories as Group Anomaly Detection problem for BERT based transformer models.
- Block Attention-anomaly Score for group anomalies among aggregated attention pattern of multiple attention heads providing transparency for model inspection.
- Extensive ablation and robustness studies addressing noise, novelties and standard deviation.

The remainder of this work is structured as follows. Section II introduces the a formal description of the addressed problem before in Section III the architecture, training and model transparency of GADFormer is proposed. The experiments in Section IV demonstrate relevance and suitability of our approach across multiple domains and Section V distinguishes our approach from related work. A final summary of the paper as well as an outline to future work is given by Section VI.

II. PRELIMINARIES AND PROBLEM DEFINITION

This section provides preliminary terminology and definitions used in this work if not referenced otherwise.

A. Preliminaries

Group Anomaly Detection (GAD) aims to identify groups that deviate from the regular group pattern[2].

Group is a set or sequence of at least two group member instances.

Group Member Instance is an arbitrary data entity described by a n-dimensional feature vector as part of a group.

[Group Anomaly or] Collective Anomaly refers to a collection of data points that belong together and, as a group, deviate from the rest of the data.[8]

B. Problem Definition

The definitions for GAD align with [2] for deep generative models, but got partially a different notation to emphasize its suitability for group anomaly detection on individual trajectories. The GAD problem is described as follows:

Let $x_n \in X$ be an instance with $X = (x_1, x_2, x_3, \dots, x_N)$ and $x_n = (a_1, a_2, a_3, \dots, a_V)$ with attribute $a_v \in \mathcal{F}$, the feature space, with

$$a_v = \begin{cases} \text{continuous, } a_v \in \mathbb{R} \\ \text{discrete, } a_v \in \mathbb{N} \\ \text{categorical, } a_v \in \{0, 1\} \end{cases}$$

Be x_{n_m} a group member instance o_i of the m th group \mathcal{G}_m with

$$\mathcal{G}_m = (o_1, o_2, o_3, \dots, o_{N_m}) \quad (1)$$

and \mathcal{D}_{GAD} a group anomaly detection dataset, which is a set \mathcal{G} of all groups:

$$\mathcal{D}_{GAD} = (\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3, \dots, \mathcal{G}_M) \quad (2)$$

The objective of the group anomaly detection task is to distinguish normal in-distribution groups from abnormal out-of-distribution groups \mathcal{G}_A with the help of a pseudo group $\mathcal{G}^{(ref)}$ as an approximated reference for normal in-distribution groups. Therefore, a characterization function f with

$$f_{\Theta} : \mathbb{R}^{N_m \times V} \rightarrow \mathbb{R}^D \quad (3)$$

and an aggregation function g with

$$g_{\phi} : \mathbb{R}^D \rightarrow \mathbb{R}^D \quad (4)$$

compose to

$$\mathcal{G}^{(ref)} = g_{\phi}(f_{\Theta}(\mathcal{G})) \quad (5)$$

where f_{Θ} maps the groups \mathcal{G}_m to D -dimensional feature vectors representing the relationship characteristics of its group members o_i and g_{ϕ} aggregates them to one D -dimensional feature vector representing one reference $\mathcal{G}^{(ref)}$ for the distribution of normal groups.

Finally, the abnormality of a group is defined by a group anomaly score y_{score} measuring the deviation by a distance measure $d(\cdot, \cdot) \geq 0$, between \mathcal{G}_m and the normal group reference $\mathcal{G}^{(ref)}$. Thus, the abnormality score y_{score} is defined as follows:

$$y_{score} = d(\mathcal{G}^{(ref)}, \mathcal{G}_m) \quad (6)$$

whereby the decision between normal and abnormal groups is defined by a threshold γ with

$$y_{label} = \begin{cases} 1, & y_{score} \geq \gamma \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

Having the group anomaly detection problem described according to [2], we also elaborate on the task of trajectory anomaly detection aligning with the notations of [3] with their slightly different problem of group trajectory anomaly detection instead of the here described individual trajectory anomaly detection:

A trajectory point p is defined as

$$p = (a_1, a_2, a_3, \dots, a_V) \quad (8)$$

A trajectory point embedding e is defined by input embedding h_1 (cf. Figure I) for each point p as

$$e = h_1(p) \quad (9)$$

Since word sentences and trajectories can both be considered as sequences we create BERT-based embeddings [15] for trajectories by defining trajectory segments s_i with e.g. $s_1 = (e_1, e_2), s_2 = (e_3, e_4), \dots$ for S segments of segment length $L_s = 2$, which represent local sequences within a trajectory. In addition to that, each segment s_i is mapped to a segment embedding h_2 , which acts as an offset for each related trajectory point embedding e_n . The sum of both, segment member e_n and segment embedding h_2 , is denoted as trajectory segment part sp with

$$sp_{i,n} = e_n + h_2(s_i) \quad (10)$$

A trajectory Λ is defined as

$$\Lambda_m = (sp_{11}, sp_{12}, sp_{23}, sp_{24}, \dots, sp_{SN_m}) \quad (11)$$

A trajectory dataset \mathcal{D}_{Traj} is defined as

$$\mathcal{D}_{Traj} = (\Lambda_1, \Lambda_2, \Lambda_3, \dots, \Lambda_M) \quad (12)$$

By considering the task of detecting abnormal individual trajectories as group anomaly detection problem, the following associations are identified:

A trajectory Λ_m applies to the semantic of a group \mathcal{G}_m by considering trajectory segments s in form of its segment parts sp as group members o . They are represented by point embeddings e adding a shared segment embedding h_2 as offset (cf. Eq. 10). The embeddings e represent trajectory points p and a trajectory point p_i is associated with an instance x_n .

Thus, individual abnormal trajectories can be detected similarly to the group anomaly detection problem (cf. Eq. 5 and Eq. 6) as follows:

$$\Lambda^{(ref)} = g_\phi(f_\Theta(\Lambda)) \quad (13)$$

$$\hat{y}_{score} = d(\Lambda^{(ref)}, \Lambda_m) \quad (14)$$

After revealing the associations between the GAD approach of [2] and our approach for trajectories, also our proposed GADFormer approach (cf. Section III) shows the potential to be trained for each arbitrary group anomaly detection problem on sequences or non-ordered sets, as far as a group to member relationship exists.

III. GADFORMER

In this section we propose GADFormer, a deep BERT based transformer encoder model architecture for attention-based Group Anomaly Detection (GAD). After we showed in Section II by the example of [2] theoretically that the GAD problem can also be applied to trajectory coordinates, we introduce in this section with GADFormer a new deep GAD model and demonstrate its performance on trajectory datasets in Section IV. Figure 2 provides an overview to its model architecture in combination with examples of 2D trajectory point inputs, but also high-dimensional group members (trajectory points) are possible.

A. Architecture and Loss Objective

Differently to the GAD characterization and aggregation function (cf. Eq. 3 and Eq. 4) of the deep generative models of [2] our deep GADFormer Ψ models the characterization and aggregation functions as follows with

$$\Psi : g_\Phi(f_\theta(\Lambda_m)) \rightarrow \hat{p}_m. \quad (15)$$

The characterization function f_θ of GADFormer maps the bidirectional relationships between group members o_i of a group \mathcal{G}_m (representations of segment parts sp of a trajectory Λ_m) to a multi-head self-attention-weight feature map $b_{\mathcal{G}_m}$, representing the behavior of an individual group (an individual trajectory path pattern). This is realized by a BERT [15] encoder, a composition of layers (h_1 and h_2) for input embedding, positional encoding and multi-head self-attention blocks (cf. Figure 2) using group member embeddings pe as input tokens. In order to extend the possible value range for an improved feature extraction, we replace the ReLU activation function of the standard FFN of [16] with Tanh.

$$b_{\Lambda_m} = f_\theta(\Lambda_m) \quad (16)$$

The aggregation function g_Φ of GADFormer approximates instead of a distribution for normal group representations $\mathcal{G}^{(ref)}$ with distance measure d a probability p_m for abnormal group behavior (abnormal trajectory path pattern). This is realized by non-linear layer blocks (2 linear projections with ReLU and a final output layer with linear compression and Sigmoid non-linearity) as part of the task output block g_Φ , which maps the group behavior characteristics b_{Λ_m} to a task specific feature map representation z_{Λ_m} .

$$z_{\Lambda_m} = g_\Phi(b_{\Lambda_m}) \quad (17)$$

After compression, the sigmoid function maps representation z_{Λ_m} to a probability \hat{p}_m for group abnormality, with $\hat{p}_m \in [0, 0.5]$ for normal groups and $\hat{p}_m \in]0.5, 1]$ for abnormal groups (trajectories).

$$\hat{p}_m = \sigma(z_{\Lambda_m}) \quad (18)$$

Because of the rare label availability for trajectories, the loss objective of GADFormer is defined for an unsupervised and semi-supervised learning setting assuming the majority of instances to be normal. Therefore, we define the binary cross entropy loss \mathcal{L}_{BCE} as our loss function (cf. Eq. 20). We consider this loss function as a suitable choice, since entropy $H(\hat{p}_m)$ as a measure of unpredictability is $H(\hat{p}_m) = 1$ when the model is most uncertain about its abnormality prediction, and $H(\hat{p}_m) = 0$ when the model is very certain about its abnormality prediction.

$$H(\hat{p}_m) = \begin{cases} 1, & \hat{p}_m = 0.5 \\ 0, & \hat{p}_m = 0 \wedge \hat{p}_m = 1 \\]0, 1[, & \text{otherwise} \end{cases} \quad (19)$$

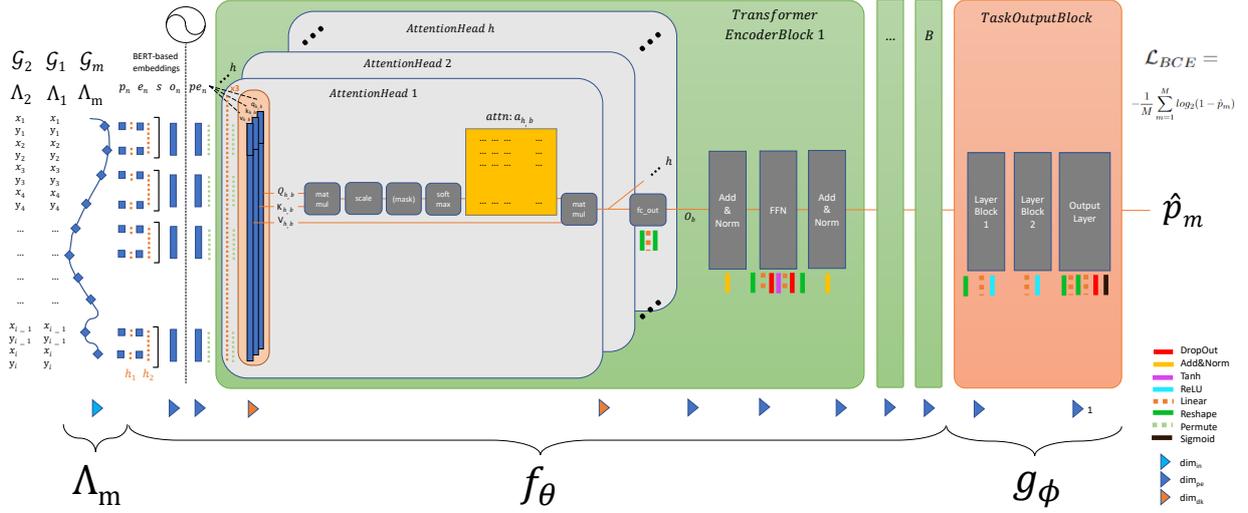


Fig. 2. GADFormer architecture overview.

Due to the heavily imbalanced learning setting with a large majority of normal groups (trajectories) one can neglect the minority of abnormal groups (trajectories) and set a fix auxiliary target probability $p_m = 0$ for certain normal-predictions ($H(\hat{p}_m) = 0$) for the majority of normal group probabilities \hat{p}_m . In case the model faces true abnormal groups, then it is rather uncertain about its decision yielding a probability close to $\hat{p}_m = 0.5$ resulting in a high entropy loss, whereas true normal groups, on whose pattern the model is trained, result in a low entropy loss for $\hat{p}_m \approx 0$.

$$\begin{aligned} \mathcal{L}_{BCE} &= \frac{1}{M} \sum_{m=1}^M p_m \log_b(\hat{p}_m) + (1 - p_m) \log_b(1 - \hat{p}_m) \\ &\stackrel{p_m=0}{\iff} \\ \mathcal{L}_{BCE} &= \frac{1}{M} \sum_{m=1}^M \log_b(1 - \hat{p}_m) \end{aligned} \quad (20)$$

Since in our setting the model effectively predicts only abnormality probabilities for the range of normal groups with $\hat{p}_m \in [0, 0.5]$, where $\hat{p}_m = 0$ means that a group (trajectory) is not abnormal at all, the abnormality of a group is defined by a group anomaly score $\hat{y}_{score} = \hat{p}_m$ for our GADFormer approach.

B. Training

Anomalous trajectories are rare by definition and labeling by domain experts tends to be rather expensive. We address this challenge by two different learning settings which are: 1) Unsupervised learning, which requires no labels at all under the assumption that the ratio of anomalous trajectories is low and has no remarkable influence during model training. 2) Semi-Supervised Learning, which relies on verified normal samples

only. As proposed in the section before, these learning settings allow us to set a fix auxiliary target probability $p_m = 0$, so that no ground truth for abnormal trajectories is needed for the GADFormer training. In order to let f_θ learn expressive representations for g_ϕ we start the training with frozen task layers, which get unfrozen as soon as validation loss stops decreasing. Furthermore, we use early stopping, learning rate scheduling ReduceOnPlateau and RAdam for optimization. Please see our supplementary material¹ for further details.

C. Model Transparency

Deep learning models are known to be rather complex and their training usually requires a deep understanding for its model architecture, losses, preprocessing and data distributions to take the right decisions for fine-tuning, but still then it partially remains a blackbox as more layers, blocks and parameters exist.

In order to achieve a higher model transparency addressing CH6, one of the main deep anomaly detection challenges of [18], we introduce a Block Attention-anomaly Score (BAS) for our GADFormer model. BAS can be seen as a further interpretable explainer for Model Inspection, solving a so called Open-The-Box-Problem[19], which allows to indicate how each layer of the transformer encoder model contributes to distinguish inputs of different ground truth classes. Class-overlapping scores in the final layer are a potential indicator for false positives and negatives respectively. Hence, BAS enables for model inspection with the goal of identifying optimization potential in the model architecture using attention matrix scores deviating from the attention matrix score mean without plausible correlation to its ground truth and neighboring ground truths. BAS follows the assumption that

¹<https://github.com/lohrrera/gadformer>

in case of the aggregated attention of a group of layer heads is anomalous then also the model input, in our case the group member instances of a trajectory, is anomalous. Considering the example of Figure 3 for a good model performance, this assumption holds for the majority of abnormal inputs across nearly all layers, especially for the final layer in which the amount of false positives decreases.

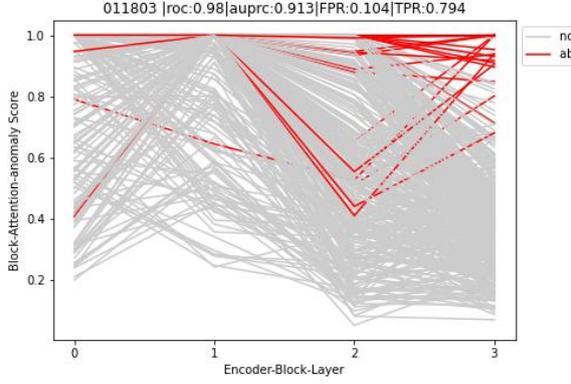


Fig. 3. BAS in case of good model performance.

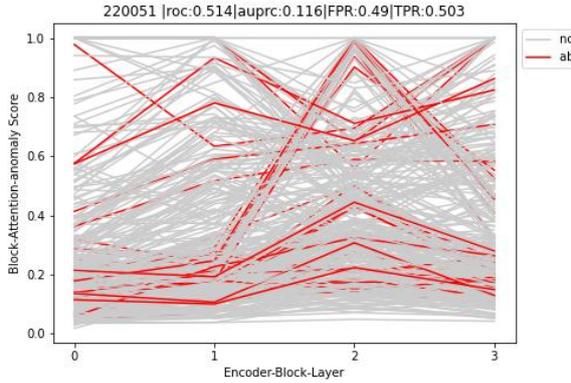


Fig. 4. BAS in case of bad model performance.

BAS represents in a transformer encoder block layer a multi-head-attention group anomaly by the ratio between distance of an aggregated block attention matrix $a_{m,b}$ and its normal block average a_b and distance of a_b to the average of $topN$ abnormal aggregated attention matrices $a_{topN,b}$ (cf. line 10 of Algorithm 1). This allows to show the capability of a transformer encoder block layer to 1) generally distinguish pattern of different groups (trajectories) and 2) to separate between normal and abnormal groups of attention head weights for individual trajectories (groups). Therewith BAS is different to the work of [20], which aims to identify single feature-relevant attention heads by maximum attention weights and histograms instead of using average attention distances.

Since the cells of an attention matrix $a_{m,h,b}$ contain the scaled dot product of $q_{m,h}$ and $k_{m,h}$ (two projected group embeddings projected from input tokens pe , cf. Figure 2), their similarity weights the importance of the bidirectional relationship of a group member pair in different heads h and

with that, focuses with different views to the behavioral pattern of a group (trajectory path pattern in the context of the task of GAD on individual trajectories), whereas its concatenated projected dot product with the third projected group member embeddings $v_{m,h}$ provides the overall-importance-weighted attention output matrix $O_{m,b}$ emphasizing task relevant pattern of a concrete group m (task relevant trajectory path segments of an individual trajectory Λ_m).

After emphasizing the role of the attention mechanism, we describe in Algorithm 1 how to calculate BAS in detail. The inputs of this algorithm are the attention matrices a of the transformer encoder blocks and the euclidean distance measure. Further parameters are the ratio for the top N abnormal groups, block index b and the amount of groups M .

Algorithm 1 BAS Algorithm Pseudo Code

Input: group attention matrices a , distance measure d

Parameters: $ratio_{topN} = 0.05$, block b , groups M

Output: block attention-anomaly scores bas

```

1:  $a_{m,b} = \mu(a_{m,h,b})$ 
2: if training then
3:    $tmp\_a_b = \mu(a_{m,b})$ 
4:    $topN = \lceil ratio_{topN} * M \rceil$ .
5:    $idx\_a_b = rank(d(a_{m,b}, tmp\_a_b), topN, dsc)$ 
6:    $idx\_n_b = idx\_all \setminus idx\_a_b$ 
7:    $global\ a_{topN,b} = \mu(a_{m,b}[idx\_a_b, :, :])$ 
8:    $global\ a_b = \mu(a_{m,b}[idx\_n_b, :, :])$ 
9: end if
10:  $bas_{m,b} = \min(1., \frac{d(a_{m,b}, a_b)}{d(a_{topN,b}, a_b)})$ 
11: return  $bas$ 

```

The average attention matrix $a_{m,b}$ for a group m in heads h is calculated for all attention matrices $a_{m,h,b}$ (line 1). During training also their temporary average tmp_a_b over all groups is calculated (line 3) to obtain their $topN$ most distant abnormal group indices idx_a_b (line 5), whose difference to all indices result in the remaining normal group indices idx_n_b (line 6). Based on these indices a global average attention heads mean for normal (a_b) and abnormal ($a_{topN,b}$) head attention averages is calculated (line 7-8) during training in order to have a solid normal and abnormal representation for distance calculation. Next, the distance between a group attention matrix $a_{m,b}$ to the normal group attention matrix average a_b as well as the distance between the normal group attention matrix average a_b and the abnormal group attention matrix average $a_{topN,b}$ is used to request the ratio between both which represents the Block Attention-anomaly Score $bas_{m,b}$ (line 10-11). Figure 3 shows, that the first encoder block layers (0-2) are not able to distinguish between normal and abnormal trajectories whereas in the last layer the amount of potential false positive scores gets less indicating a better capability of the model to attend to features of abnormal trajectories. The BAS within the layers of Figure 4 indicate that the model is over all layers not really able to distinguish between normal and abnormal trajectory scores, not even between the characteristics of single trajectories within the class of normal trajectories.

In summary, the BAS provides us a view to the transformer encoder block layers for model inspection and allows us to reason reasonable changes to hyperparameters and model architecture to improve the performance of the model. Providing a further answer to "Do Transformer Attention Heads Provide Transparency in Abstractive Summarization?"[20], we are able to provide attention block transparency in terms of which degree the averaged attention of a group of self-attention-heads within one layer is normal or abnormal.

IV. EXPERIMENTS

In this section we evaluate the performance of our GADFormer approach on synthetic and real-world datasets and compare it against related works like GRU and MainTulGAD, whose approaches are state of the art methods for individual trajectory anomaly detection. For details related to datasets, architectures, hyperparameters, results and code see our supplementary material¹.

A. Experimental Setup and Datasets

For our experiments we tested our approach on synthetic data¹ and three real-world datasets, i.e., amazon driving routes², Deutsche Bahn cargo container routes³ and brightkite checkin routes⁴ (cf. Table I). The synthetically generated trajectory dataset consists of trajectory steps, one per row, where each has an id, sequence step, xcoord, ycoord, and a label whether its trajectory (=group) is anomalous (1) or normal (0).

All hyperparameters are empirically selected by grid search based on validation loss convergence and additionally validated by model inspection with our proposed block attention anomaly score (cf. BAS Algorithm 1) in order to find ideal training parameters and a model architecture avoiding overfitting or insufficient model complexity.

The code for GADFormer is implemented in Python utilizing PyTorch. For best possible comparison, the architecture for GRU is identical to GADFormer except using GRU-layers instead of encoder block attention layers and only input embedding (cf. Eq. 9) instead of BERT segmentation. Our MainTul [17] version (MTGAD) uses kNN-trajectory-augmentations and a student-teacher-architecture for feature

²<https://github.com/amazon-science/goal-gps-ordered-activity-labels>

³<https://data.deutschebahn.com/dataset/data-sensordaten-schenker-seefrachtcontainer.html>

⁴<https://snap.stanford.edu/data/loc-brightkite.html>

TABLE I
DATASET OVERVIEW.

dataset	setting	all	normal	abnormal	trajLen
synthetic ¹	unsup	3400	3083	317	72
synthetic ¹	semi	3400	3271	129	72
amazon ²	unsup	805	760	45	72
amazon ²	semi	776	760	16	72
dbcargo ³	unsup	272	229	43	72
dbcargo ³	semi	245	229	16	72
brightkite ⁴	unsup	2241	2033	208	500
brightkite ⁴	semi	2108	2033	75	500

extraction like the original, but adapts to sequential trajectory coordinates instead of time-dependent categorical checks. These approaches represent the technically most related work for a fair comparison. Extended results with less related traditional methods can be found in supplementary material¹.

B. Evaluation

For the evaluation of our model, we follow the goals of having a low miss rate (false negatives) as well as achieve as less false alerts (false positives) as possible. In addition to that, the quality of the model scores needs to be evaluated. Therefore and to be comparable to related approaches, we evaluate the model performance by AUROC and AUPRC.

- $TPR = \frac{TP}{P}$; $FPR = \frac{FP}{N}$; $FNR = 1 - TPR$
- Precision = $\frac{TP}{TP+FP}$; Recall = $\frac{TP}{TP+FN}$
- AUPRC = AP = Precision vs. Recall
- AUROC = TPR vs. FPR

C. Results and Discussion

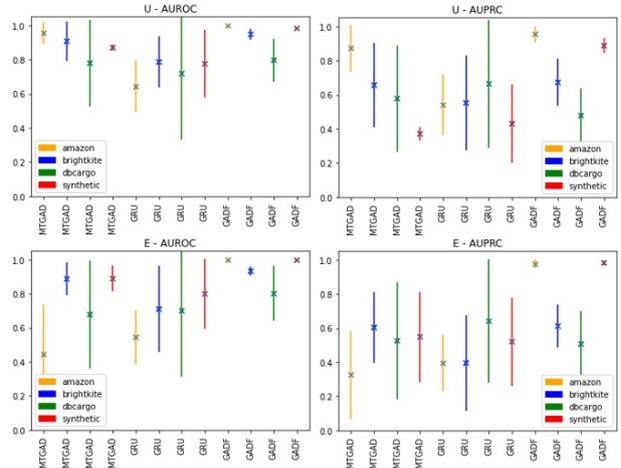


Fig. 5. Robustness results for synthetic and real world datasets of Table II. Span shows the stddev of 10 seeds; cross is the used mean performance.

Our approach GADFormer (GADF) outperforms related works GRU and MainTulGAD (MTGAD) on all synthetic and real world datasets in terms of AUROC and AUPRC except for AUPRC on dbcargo for which an approach like GRU could achieve a better performance for both un- and

TABLE II
RESULTS ON SYNTHETIC AND REAL WORLD DATASETS FOR (U)NSUPERVISED- AND S(E)MI-SUPERVISED SETTING (CF. FIGURE 5).

	dataset	amazon		brightkite		dbcargo		synthetic	
		auroc	auprc	auroc	auprc	auroc	auprc	auroc	auprc
U	GRU	0.642	0.539	0.786	0.552	0.718	0.664	0.775	0.431
	MTGAD	0.956	0.872	0.907	0.656	0.779	0.577	0.87	0.371
	GADF	0.997	0.955	0.948	0.672	0.797	0.478	0.982	0.887
E	GRU	0.545	0.394	0.711	0.396	0.701	0.64	0.799	0.52
	MTGAD	0.445	0.325	0.887	0.604	0.678	0.526	0.889	0.549
	GADF	0.998	0.976	0.933	0.612	0.801	0.507	0.997	0.982

TABLE III
RESULTS ON SYNTHETIC DATASET WITH NOISE ABLATIONS FOR
(U)NSUPERVISED AND S(E)MI-SUPERVISED SETTING

exp		noise .0		noise .2		noise .5	
		auroc	auprc	auroc	auprc	auroc	auprc
U	GRU	0.766	0.514	0.731	0.383	0.626	0.165
	MTGAD	0.869	0.376	0.822	0.256	0.717	0.149
	GADF	0.97	0.892	0.949	0.831	0.863	0.537
E	GRU	0.788	0.585	0.759	0.479	0.665	0.223
	MTGAD	0.952	0.766	0.89	0.547	0.792	0.316
	GADF	0.989	0.95	0.98	0.919	0.944	0.803

TABLE IV
RESULTS ON SYNTHETIC DATASET WITH NOVELTY ABLATIONS FOR
(U)NSUPERVISED AND S(E)MI-SUPERVISED SETTING.

exp		novelty .0		novelty .01		novelty .05	
		auroc	auprc	auroc	auprc	auroc	auprc
U	GRU	0.766	0.514	0.832	0.585	0.818	0.496
	MTGAD	0.882	0.42	0.935	0.588	0.923	0.504
	GADF	0.97	0.892	0.978	0.865	0.969	0.726
E	GRU	0.788	0.585	0.849	0.652	0.841	0.574
	MTGAD	0.964	0.802	0.977	0.867	0.97	0.797
	GADF	0.989	0.95	0.986	0.921	0.986	0.841

semisupervised settings. Considering Figure 5 and Table II, GADFormer demonstrates its stability across all datasets with lowest standard deviations over 10 seeds. For datasets amazon, brightkite and synthetic its AUROC standard deviation is close to zero. AUROC performances over 0.8 on real-world datasets in semisupervised settings highlight its relevance for real-world-domains, especially for amazon routes for which it achieved performances over 0.95 for all metrics. Also on brightkite (a dataset with long sequences of 500 steps) showed our transformer-based approach still the best performance, demonstrating its superiority against GRU and MTGAD which both are at least partially based on recurrent neural networks. MTGAD with its self-supervised augmented kNN-trajectories and its combined student-teacher-approach of LSTM and multi-head attention shows performances close to but slightly weaker than GADFormer except for AUPRC of dbcargo. In ablation studies for detecting noise-distorted and novel anomalies shows GADFormer a comparable strong performance even for high noise and novelty ratios from 0 up to 0.5 or 0.05 respectively (cf. Table III and Table IV). Summarizing, compared to related work, GADFormer (GADF) can be considered as a robust approach, but despite its strong false and miss alert rates (evaluated via AUROC and AUPRC) it depends on the domains if these performances are sufficient.

V. RELATED WORK

Reviewing the literature for most related approaches, we could identify the following related work, which gets distinguished from our approach within this section. Instead of considering the detection of individual trajectory anomalies as a Group Anomaly Detection problem as our approach does, the vision in the works of [21, 22] is to observe trajectories as a NLP problem. They map trajectory coordinates to hexagon-based hexadecimal-words as input for pretrained BERT mod-

els for several tasks, but do not provide a concrete model architecture for group anomaly detection based on projected trajectory segments as BERT-based embeddings. Another work of [17] uses for the task of Trajectory-User-Linking (TUL), instead of GAD, a combination out of RNN and transformer network with cross entropy loss but compared to our approach, they do not take trajectory coordinates and segments into account and the model lacks in layer transparency. Addressing long-range trajectory anomaly detection as well the work of [23] proposes an unsupervised normalizing flow (NF) model. They utilize trajectory segments and negative log-likelihood as well but use it in combination with NF-based density estimation. The work of [24] introduced the problem of group trajectory outlier detection (GTOD), which is also addressed by [3], and provide the approach CDkNN, which creates DBSCAN-based microclusters, pruned by kNN and scored with a specific pattern mining algorithm. However, both works perform anomaly detection while considering complete individual trajectories as group members, whereas we address the slightly different problem of considering single trajectory points as group members for the problem group anomaly detection. The work of [25] proposes a model for content-aware anomaly detection on event log messages instead of anomalous trajectories as our approach. Their approach takes additionally the content of the messages into account and allows to run it, as our approach, by the task-specific encoder-part or, differently as ours, by the typical BERT[15] encoder-decoder architecture. Summarizing the identified related work, there is best to our knowledge no transparent attention-based transformer-encoder-approach for group anomaly detection on coordinates-based trajectories.

VI. CONCLUSION

In this work we proposed GADFormer, a transformer-encoder-architecture, capable to perform attention-based group anomaly detection in an unsupervised and semi-supervised setting. We emphasized, how the detection of individual anomalous trajectories can be solved as a Group Anomaly Detection (GAD) problem for BERT based transformer models. Furthermore, we introduced BAS, a Block Attention-anomaly Score to allow model inspection for transformer encoder blocks for the task of GAD and improve with that its transparency in terms of answering to which degree the attention of the group of self-attention-heads is normal or abnormal. Extensive ablation and robustness studies addressing trajectory noise and novelties on synthetic and real world datasets demonstrated, that our approach is on par with related attention-based approaches like GRU. Further potential for improvement could be to approximate a normal-group-distribution instead of abnormal-group-probabilities by the output-block of our model, combining the attention-based group pattern extraction of our approach and the group anomaly scoring and loss objectives of [2]. Vice versa, with appropriate preprocessing, the performance of the GADFormer model architecture could also be evaluated on image data, audio or text data. Moreover, the reliability of the probabilities of our approach could be further investigated

according to the work of [26] as well as the relevance of single group member instances for a specific model prediction.

REFERENCES

- [1] Liang Xiong, Barnabás Póczos, and Jeff Schneider. “Group Anomaly Detection Using Flexible Genre Models”. In: *Proceedings of the 24th International Conference on Neural Information Processing Systems. NIPS’11*. Granada, Spain: Curran Associates Inc., 2011, pp. 1071–1079.
- [2] Raghavendra Chalapathy, Edward Toth, and Sanjay Chawla. “Group Anomaly Detection Using Deep Generative Models”. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part I*. Vol. 11051. Springer, 2018, pp. 173–189.
- [3] Asma Belhadi et al. “Machine Learning for Identifying Group Trajectory Outliers”. In: *ACM Trans. Manage. Inf. Syst.* 12.2 (Jan. 2021).
- [4] Andreas Lohrer, Johannes Josef Binder, and Peer Kröger. “Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities”. In: *Proceedings of the 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science. IWCTS ’22*. Seattle, Washington: Association for Computing Machinery, 2022.
- [5] Krikamol Muandet and Bernhard Schölkopf. “One-Class Support Measure Machines for Group Anomaly Detection”. In: *UAI’13*. Bellevue, WA: AUAI Press, 2013, pp. 449–458.
- [6] Benjamin Nachman and David Shih. “Anomaly detection with density estimation”. In: *Phys. Rev. D* 101 (7 Apr. 2020), p. 075042.
- [7] Rose Yu, Xinran He, and Yan Liu. “GLAD: Group anomaly detection in social media analysis”. In: 2014.
- [8] Ralph Foorthuis. “On the nature and types of anomalies: a review of deviations in data”. In: *International Journal of Data Science and Analytics* 12 (2020), pp. 297–331.
- [9] Zhongqiu Wang et al. “Unsupervised learning trajectory anomaly detection algorithm based on deep representation”. In: *International Journal of Distributed Sensor Networks* 16 (2020).
- [10] Xiang Jiang et al. “Improving point-based AIS trajectory classification with partition-wise gated recurrent units”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. 2017, pp. 4044–4051.
- [11] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780.
- [12] Kyunghyun Cho et al. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. 2014.
- [13] Liang Xiong et al. “Hierarchical Probabilistic Models for Group Anomaly Detection”. In: *International Conference on Artificial Intelligence and Statistics*. 2011.
- [14] Bernhard Schölkopf et al. “Estimating the Support of a High-Dimensional Distribution”. In: *Neural Computation* 13.7 (2001), pp. 1443–1471.
- [15] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019, pp. 4171–4186.
- [16] Ashish Vaswani et al. “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS’17*. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010.
- [17] Wei Chen et al. “Mutual Distillation Learning Network for Trajectory-User Linking”. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. International Joint Conferences on Artificial Intelligence Organization, July 2022, pp. 1973–1979.
- [18] Guansong Pang et al. “Deep Learning for Anomaly Detection: A Review”. In: *ACM Comput. Surv.* 54.2 (Mar. 2021).
- [19] Riccardo Guidotti et al. “A Survey of Methods for Explaining Black Box Models”. In: *ACM Comput. Surv.* 51.5 (Aug. 2018).
- [20] Joris Baan et al. “Do Transformer Attention Heads Provide Transparency in Abstractive Summarization?” In: *ArXiv abs/1907.00570* (2019).
- [21] Mashaal Musleh, Mohamed F Mokbel, and Sofiane Abbar. “Let’s speak trajectories”. In: *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. 2022, pp. 1–4.
- [22] Mashaal Musleh. “Towards a unified deep model for trajectory analysis”. In: *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. 2022, pp. 1–2.
- [23] Madson L. D. Dias et al. “Anomaly Detection in Trajectory Data with Normalizing Flows”. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. 2020, pp. 1–8.
- [24] Youcef Djenouri et al. “Fast and Accurate Group Outlier Detection for Trajectory Data”. In: *New Trends in Databases and Information Systems*. Cham: Springer International Publishing, 2020, pp. 60–70.
- [25] Shengming Zhang et al. “CAT: Beyond Efficient Transformer for Content-Aware Anomaly Detection in Event Sequences”. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, pp. 4541–4550.
- [26] Zhengbao Jiang et al. “How Can We Know What Language Models Know?” In: *Transactions of the Association for Computational Linguistics* 8 (2019), pp. 423–438.

Chapter 6

Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities

This chapter consists of a preprint version of the following publication:

Andreas Lohrer, Johannes Josef Binder, and Peer Kröger. “Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities”. In: *Proceedings of the 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science, IWCTS 2022, Seattle, Washington, 1 November 2022*. Ed. by Andy Berres, Kuldeep R. Kurte, and Haowen Xu. ACM, 2022, 14:1–14:4. DOI: 10.1145/3557991.3567801. URL: <https://doi.org/10.1145/3557991.3567801>

- **Conception:** Lohrer (Lead)
- **Planning:** Lohrer (Lead)
- **Execution:** Lohrer (Lead), Binder (Support)
- **Manuscript:** Lohrer (Lead), Binder (Support), Kröger (Support)

Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities

Andreas Lohrer
Ludwig-Maximilians-Universität
München
München, Bavaria, Germany
Christian-Albrechts-Universität zu
Kiel
Kiel, Schleswig-Holstein, Germany
alo@informatik.uni-kiel.de

Johannes Josef Binder
Ludwig-Maximilians-Universität
München
München, Bavaria, Germany
johannes.binder@campus.lmu.de

Peer Kröger
Christian-Albrechts-Universität zu
Kiel
Kiel, Schleswig-Holstein, Germany
pkr@informatik.uni-kiel.de

ABSTRACT

Group anomaly detection in terms of detecting and predicting abnormal behaviour from entities as a group rather than as an individual, addresses a variety of challenges in spatio-temporal environments like e.g. traffic and transportation systems, smart cities, geoinformation systems, etc. They provide information about a commonly large number of individual entities. Examples for such entities would be airplanes and drones, vehicles, ships but also people, remote sensors and any other information source in interaction with the environment. However, as point anomaly detection is quite common for revealing the abnormal behaviour of individual entities, the collective behaviour of the individuals as a group remains completely uncovered. For example potential for traffic flow optimizations or increased local traffic guideline violations cannot be detected by one single drive but by considering the behavior of a group of vehicle drives in this area. With this work-in-progress we elaborate the potential of group anomaly detection algorithms for spatio-temporal collective behaviour scenarios in smart cities. We describe the group anomaly detection problem in the context of urban planning and demonstrate its effectiveness on a public real-world data set for urban rental bike rides and stations in and around Munich revealing abnormal groups of rides, which allows to optimize the rental bike accessibility to the population and with that to contribute to a sustainable environment.

CCS CONCEPTS

• **Computing methodologies** → **Anomaly detection; Search methodologies; Machine learning approaches**; • **Information systems** → **Spatial-temporal systems**.

KEYWORDS

group anomaly detection, collective anomaly detection, smart cities, urban planning, machine learning, artificial intelligence

ACM Reference Format:

Andreas Lohrer, Johannes Josef Binder, and Peer Kröger. 2022. Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities. In *The 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science (IWCTS '22)*, November 1, 2022, Seattle, WA, USA. ACM, New York, NY, USA, 4 pages. <https://doi.org/10.1145/3557991.3567801>

1 SPATIO-TEMPORAL COLLECTIVE BEHAVIOR SCENARIOS

In times of an increasingly growing world population regions and living areas without information providing people, species, devices and sensors are becoming rare. Burdening the planet on the one hand-side spatially exhaustive and continuously available information enables for sustainable counter-steering measures like environmental monitoring, earth observation or other resources optimizing initiatives like smart cities, intelligent transportation systems and others. What all these application areas have in common, is the fact that they can support many of their goals effectively by considering information from individuals as a group instead of, or additionally to, as a single entity. The information of entities as members of a group represents together a collective behavior, which can be further differentiated as normal or abnormal group behavior. In spatio-temporal environments there are various scenarios in which abnormal group behaviors are desired to be uncovered. The following examples provide a short overview of groups with potential abnormal behavior. Images of human crowds like travelers, tourists, event visitors, etc., which are moving or behaving differently as expected can disclose infrastructure capacity shortages or pandemic developments after contact tracking. In the wildlife, remote sensors and images of herds, swarms or packs of animals or organisms can abnormally share or visit unexpected areas at unusual times, move together, etc. which allows to infer populations, living area circumstances or species interactions. Furthermore, GPS and operation sensors from fleets of airplanes, vehicles or ships showing abnormal fuel, energy or wearing part consumption can provide insights about route environments, changes according to driving guidelines or routes. Considering economical domains, the group behavior of salesmen, suppliers and customers (positioning, trajectories, location-based sales or purchases, etc.) at a shop, market or trade fair hall or at any other arbitrary urban region has the potential to reveal abnormal positive or negative environmental influences for purchases or sales. In logistic or grocery manufacturing locations allow RFID chips and other remote sensors to uncover

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IWCTS '22, November 1, 2022, Seattle, WA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9539-7/22/11...\$15.00
<https://doi.org/10.1145/3557991.3567801>

global issues related to maintenance, supplier quality or operation and hence to avoid negative impacts to the urban supply chain.

Trying to detect abnormal behavior as described in the aforementioned scenarios, traditional point anomaly detection algorithms would tend to fail since they are only classifying and scoring single instances as anomalies and neglect to consider behavioral dependencies within the group context.

For modeling and detecting group behavior in static and dynamic scenarios the framework of Toth et al.[8] refers to three sub-problems: 1) definition of group structures and relationships, 2) quantification of statistical properties and 3) change/deviation detection and scoring e.g. by hypothesis tests, discriminative or generative models. Group structures are either known upfront or obtained after clustering. In a subsequent step groups can be scored based on properties like density, orientation or shape, and also in combination with given contexts, e.g. time, location or other contextual subspaces.

Differently to statistical testing (e.g. [9]), established svm-based methods (e.g. [6], [7]) or deep learning approaches (e.g. [1], [5], etc.) learn group representations and score groups based on given distance measures or loss functions to distinguish between normal and abnormal groups.

In this work-in-progress we elaborate the potential of group anomaly detection (GAD) approaches for spatio-temporal collective behaviour scenarios in smart cities. We distinguish between instance- and distribution-based group anomaly detection, describe the group anomaly detection problem in the context of urban planning and demonstrate its effectiveness in scope of a case study on real-world mobility information.

The contributions of this work-in-progress can be summarized as follows:

- introduction of GAD as machine learning approach for spatio-temporal collective behavior detection in smart cities
- case study: distribution-based GAD as optimization approach for urban planning
- public real-world dataset for GAD on spatio-temporal mobility information of urban bike rentals

The remaining chapters of this paper are Section 2 introducing the GAD preliminaries, followed by Section 3 demonstrating the potential of distribution-based GAD for urban planning in smart cities by a case study before Section 4 describes experiments and results. The paper summarizes with a conclusion in section 5.

2 PRELIMINARIES

In this section we describe the differences between point-based and distribution-based group anomalies (cf. Foorthuis [3] types VII-i and VII-j) and provide a common understanding for the definition of the group anomaly detection problem.

2.1 Point- and Distribution-based Group Anomaly Detection

Chalapathy et al.[1] describe a "point-based anomalous group [as] a collection of individual pointwise anomalies that deviate from the expected pattern". In contrast, a point-wise anomaly is considered as a single abnormal data point or outlier. A further definition of

Xiong et al. [10] describes a "point-based group anomaly [as] a group of individually anomalous points". These are also known as Micro-Clusters. Differently to point-based group anomalies the distribution-based anomalies, whose detection is of main interest in scope of this work to optimize urban planning, allow also normal points as part of abnormal groups. In the work [1] Chalapaty et al. mention that "distribution-based group anomalies [...] are seemingly regular however their collective behavior is anomalous". In addition to that the work of Xiong et al. [10] describes "a distribution-based anomaly [as] a group where the points are relatively normal, but as a whole they are unusual". As an example for type VII-j), Figure 2 shows in red the distributions for our case study.

2.2 Group Anomaly Detection Problem

For a better understanding of the common Group Anomaly Detection Problem a formal definition is provided in the following according to the notion of Chalapaty et al. [1] and Kuppa et al.[5]:

Group Anomaly Detection requires groups $\mathcal{G} = \{G_m\}_{m=1}^M$ where the m th group contains N_m instances with

$$G_m = (X_{nv}) \in \mathbb{R}^{N_m \times V} \quad (1)$$

where X_{nv} is the v th feature ($v = 1, 2, \dots, V$) of instances n ($n = 1, 2, \dots, N_m$) in the group G_m , \mathbb{R} is a continuous value domain. The total number of individual instances is $N = \sum_{m=1}^M N_m$. In GAD the behavior or properties of the m th group is captured by a characterization function denoted by $f : \mathbb{R}^{N_m \times V} \rightarrow \mathbb{R}^D$ where D is the dimensionality on the transformed feature space.

After a characterization function is applied to a training dataset, group information is combined using an aggregation function $g : \mathbb{R}^{M \times D} \rightarrow \mathbb{R}^D$

A group reference is composition of characterization and aggregation functions on the input groups with

$$\mathcal{G}^{(ref)} = g[\{f(G_m)\}_{m=1}^M] \quad (2)$$

Then a distance metric $d(\cdot, \cdot) \geq 0$ is applied to measure the deviation of a particular group from the group reference function. The distance score $d(\mathcal{G}^{(ref)}, G_m)$ quantifies the deviance of the m th group from the expected group pattern where larger values are associated with more anomalous groups.

3 CASE STUDY: DISTRIBUTION-BASED GAD AS OPTIMIZATION APPROACH FOR URBAN PLANNING

3.1 Domain and Motivation

Smart cities[2] are commonly defined by the six aspects people, living, economy, mobility, environment and governance. This case study has its focus especially on the mobility aspect with the goal of demonstrating the effectiveness of distribution-based group anomaly detection for revealing optimization potential for urban planning.

In this scope the study investigates in the optimization of rental bike rides and returns in the area of Munich. The Munich Transport Association (MVG)¹ offers with "MVG Rad"² a bike rental service

¹<https://www.mvg.de>

²<https://www.mvg.de/services/mvg-rad.html>

to the population allowing its users to rent a bike from a pool of 4500 bikes by booking it via the related service app. According to the MVG rental guidelines³ there is a differentiation between free and non-free return regions⁴ (cf. Figure 1). In free return regions users are allowed to return the bike at one of the 300 bike rental stations but also at well visible self-selected places. In non-free return regions bike returns are only allowed at a bike rental station. Otherwise the user gets charged with an extra fee for the return of the bike by MVG. Although these guidelines reasonably care for service accessibility the question arises if there is further potential for improvement considering the actual spatial rental bike ride and return behavior of different bike ride groups.

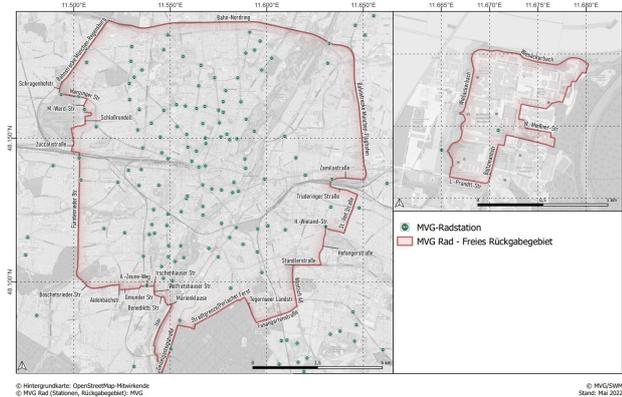


Figure 1: Map of Munich illustrating the borders between non-free and free return regions (free inside the polygons) with the city of Munich on the left and the free return region satellite of TU Munich on the right.⁴

In order to demonstrate the effectiveness of distribution-based group anomaly detection for revealing optimization potential in the scope of public rental bike rides for which there is usually no ground truth information about actual group anomalies available the following research questions arise:

- RQ1 How is it possible to detect abnormal groups of bike rides by distribution-based GAD with unsupervised machine learning methods?
- RQ2 How is it possible to define meaningful group structures for distribution-based GAD on bike rides?
- RQ3 How to model normal and abnormal group behavior for distribution-based GAD on bike rides?
- RQ4 How can GAD model predictions of abnormal bike ride groups be used to optimize urban planning?

For investigation in these research questions the study uses the continuously updated and publicly available real-world datasets provided at MVG Rad website². These MVG Rad datasets contain bike ride information for rental start and end like time, GPS location (longitude and latitude), rental and return bike station name (e.g. Josephsplatz) and flag (GPS location matches station location

³<https://www.mvg.de/services/mvg-rad/mvg-rad-agb.html>

⁴<https://www.mvg.de/dam/jcr:d8d39828-995f-4d6a-892b-7466d041ab3b/ge-schaeftsgebiet-mvg-rad.pdf>

yes or no) for the years 2015 to 2021 - in total 3342060 bike rides. All information provided ensures the anonymity of the bicyclists.

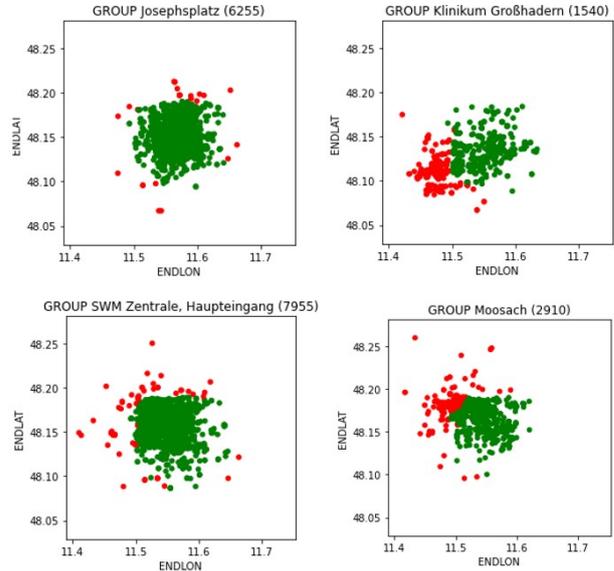


Figure 2: Illustration of normal return region ride distributions on the left and abnormal distributions on the right. (considering only returns of red non-free return region distributions and excluding green free return regions)

In this domain (group) behavior is represented by transportation activities of GPS equipped rental bikes providing its location only from the start and end location of a rental bike ride together with the related date and time. For this study this information is the foundation to learn patterns of spatio-temporal behavior.

3.2 Methodological Approach

Before distribution-based GAD can be applied in the context of urban planning the GAD problem is required to be defined accordingly to the given problem domain of rental bike rides.

Definition of groups and group members: Empirical analysis revealed that there are differences in the distributions of bike ride end locations (cf. Figure 2) of bike rides with specific rental bike stations as start location with station name as GID. Thus transportation activity groups can be represented by rental bike stations as start location whereas the corresponding bike rides shape the group as group members. The behavior of a member of a group \mathcal{G}_m is described by $x_m \in \mathbb{R}^2$ with $x_m = (x_{EndLon}, x_{EndLat})$ as feature vector. In order to represent the collective behavior of all group members the covariance matrix Σ_m is calculated for all group members x_m and transformed as group feature vector $x'_m = (\sigma_{11_m}, \sigma_{12_m}, \sigma_{21_m}, \sigma_{22_m})$ with $x'_m \in \mathbb{R}^4$ representing the covariance of the distributions of GPS locations where the bike got returned. (RQ2)

Definition of normal and abnormal group behavior: Although different covariances are likely to get distinguished by distribution-based GAD methods they have semantically no meaning for optimizations in urban planning. Considering distributions of rental

stations (start locations) of free and non-free return regions one recognizes that the majority of rentals from free return regions also end in free return regions (cf. Josephsplatz or SWM Zentrale in Figure 2). According to the domain guidelines that in non-free regions bike returns are only allowed at bike stations the variance of return coordinates is expected to be very low. Nevertheless there are rental bike stations in non-free return areas which show frequent bike rides with high variance return coordinates of frequently visited points of interest (cf. Figure 2 with e.g. Moosach, Klinikum Großhadern), which is normal behavior from user perspective but abnormal behavior according to the domain guidelines. Rental stations with high variance bike rides to rarely visited points of interest remain considered as normal. (RQ3)

Model Selection for distribution-based GAD: Due to the non-availability of verified labeled group samples we choose an unsupervised machine learning approach to distinguish between normal and abnormal group distributions. Related but different to the One-class support measure machine (OCSMM)[6], introduced as generalization of One-Class Support Vector Machine (OCSVM)[7] for distribution-based GAD, we formulate the group behavior by the covariance matrix Σ_m for group member feature vectors x as described in the RQ2 paragraph above. The group feature vectors x'_m act as reshaped covariance matrices as training samples representing group distributions by following the top-down approach as well. In order to detect abnormal bike ride group distributions with high variant but frequently appearing end location coordinates our method of choice is OCSVM in combination with our covariance-based group feature vector x'_m and with the radial base function (RBF) as non-linear kernel method. (RQ1)

4 EXPERIMENTS

4.1 Experimental Setup and Evaluation

For evaluation of our approach we use the MVG Rad dataset introduced in section 3.1 for the years 2015 to 2018 (1247057 samples) and pseudo labels based on an empirically selected threshold ($1e-5$) for group size weighted absolute covariance sums. As stated in section 3.2 for RQ3 only bike rides of rental stations from non-free return regions are relevant to revealing optimization potential for urban planning. This and further filtering of invalid zero coordinates reduces the data from 1247057 rides from 329 groups to 14183 rides from 157 groups. For the hyperparameter search of the critical OCSVM parameter γ a grid search has been applied resulting in an ideal choice of $\gamma = 5.55$. Our approach has been evaluated based on the metrics Area under the ROC curve (AUROC) and Recall.

4.2 Result Discussion

Differently to the work of [6] which used group means as training samples for OCSVM with less success in their evaluation, we could achieve reasonable results (ROC: 0.886 Recall: 1.0) with OCSVM in combination with our covariance-based group feature vector representations as training samples for distribution-based GAD.(RQ1) The application of distribution-based GAD in combination with OCSVM[7] as unsupervised machine learning method allows to detect frequently visited non-free return regions as potential future free return region satellites to further improve the accessibility for bike rentals in the domain of urban planning. Such free return

region satellites can be public points of interest like universities (e.g. the already established free return region satellite for TU Munich - cf. Figure 1), schools, sport or shopping centers (Fürstenried West), locations of large company areas (Am Hart), stadiums (Allianz Arena), fairground areas (Messestadt West) but also churches, hospital areas (Klinikum Großhadern), settlements or other area segments which are frequently visited but without directly visible bike rental station. The improved accessibility in free return region satellites could save time and effort, and motivate for a more frequent usage of bikes instead of combustion engine based vehicles to sustainably contribute to emission reduction (RQ4).

5 CONCLUSION

In conclusion we introduced GAD as machine learning approach for spatio-temporal collective behavior detection in smart cities. In scope of a case study we demonstrated the effectiveness of distribution-based GAD as optimization approach for urban planning revealing frequently visited areas without directly visible bike stations in non-free return regions and propose for these free region satellites as optimization potential to improve the accessibility of rental bikes to the population. For future work an additional investigation in temporal group anomalies evaluating if timely restricted free region satellites are reasonable and if distribution-based GAD is also effective for other transportation services, e.g. eScooters, or for other bike datasets like in [4], might be of interest. Furthermore, the comparison of the proposed methodology with other distribution-based GAD methods might be interesting as well as the investigation in further possibilities to optimize characterization, aggregation and scoring functions for effective GAD.

ACKNOWLEDGMENTS

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

REFERENCES

- [1] Raghavendra Chalapathy, Edward Toth, and Sanjay Chawla. 2018. Group anomaly detection using deep generative models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 173–189.
- [2] O.Y. Ercoskun. 2011. *Green and Ecological Technologies for Urban Planning: Creating Smart Cities: Creating Smart Cities*. Information Science Reference.
- [3] Ralph Foorthuis. 2021. On the nature and types of anomalies: A review of deviations in data. *International Journal of Data Science and Analytics* 12, 4 (2021), 297–331.
- [4] Thomas Koch and Elenna R Dugundji. 2021. Taste variation in environmental features of bicycle routes. In *Proceedings of the 14th ACM SIGSPATIAL International Workshop on Computational Transportation Science*. 1–10.
- [5] Aditya Kuppa, Slawomir Grzonkowski, Muhammad Rizwan Asghar, and Nhien-An Le-Khac. 2019. Finding Rats in Cats: Detecting Stealthy Attacks using Group Anomaly Detection. *2019 18th IEEE International Conference on Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)* (2019), 442–449.
- [6] Krikamol Muandet and Bernhard Schölkopf. 2013. One-class support measure machines for group anomaly detection. *arXiv preprint arXiv:1303.0309* (2013).
- [7] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex Smola, and Robert C. Williamson. 2001. Estimating the Support of a High-Dimensional Distribution. *Neural Computation* 13 (2001), 1443–1471.
- [8] Edward Toth and Sanjay Chawla. 2018. Group deviation detection methods: a survey. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 1–38.
- [9] Weng-Keen Wong, Andrew Moore, Gregory Cooper, and Michael Wagner. 2003. WSARE: what's strange about recent events? *Journal of Urban Health* 80, 1 (2003), i66–i75.
- [10] Liang Xiong, Barnabás Póczos, and Jeff G. Schneider. 2011. Group Anomaly Detection using Flexible Genre Models. In *NIPS*.

Chapter 7

OAB - An Open Anomaly Benchmark Framework for Unsupervised and Semisupervised Anomaly Detection on Image and Tabular Data Sets

This chapter consists of a preprint version of the following publication:

Andreas Lohrer, Jan Deller, Maximilian Hünemörder, and Peer Kröger. “OAB - An Open Anomaly Benchmark Framework for Unsupervised and Semisupervised Anomaly Detection on Image and Tabular Data Sets”. In: *2021 International Conference on Data Mining, ICDM 2021 - Workshops, Auckland, New Zealand, December 7-10, 2021*. IEEE, 2021, pp. 991–1000. DOI: 10.1109/ICDMW53433.2021.00129

- **Conception:** Hünemörder (Lead), **Lohrer** (Lead), Deller (Support)
- **Planning:** **Lohrer** (Lead), Deller (Lead), Hünemörder (Support)
- **Execution:** Deller (Lead), **Lohrer** (Support)
- **Manuscript:** **Lohrer** (Lead), Deller (Lead), Hünemörder (Support), Kröger (Support)

OAB - An Open Anomaly Benchmark Framework for Unsupervised and Semisupervised Anomaly Detection on Image and Tabular Data Sets

Andreas Lohrer †
Information Systems and Data Mining
Kiel University
24118 Kiel, Germany
Email: alo@informatik.uni-kiel.de
ORCID: 0000-0001-7834-301X

Jan Deller †
Kiel University
24118 Kiel, Germany
Email: jandeller@t-online.de
† first authors

Maximilian Hünemörder
Information Systems and Data Mining
Kiel University
24118 Kiel, Germany
Email: mah@informatik.uni-kiel.de

Peer Kröger
Information Systems and Data Mining
Kiel University
24118 Kiel, Germany
Email: pkr@informatik.uni-kiel.de
ORCID: 0000-0001-5646-3299

Abstract—We introduce OAB, an Open Anomaly Benchmark Framework for unsupervised and semisupervised anomaly detection on image and tabular data sets, ensuring simple reproducibility for existing benchmark results as well as a reliable comparability and low-effort extensibility when new anomaly detection algorithms or new data sets are added. While making established methods of the most popular benchmarks easily accessible, OAB generalizes the task of un- and semisupervised anomaly benchmarking and offers besides commonly used benchmark data sets also semantically meaningful real-world anomaly data sets as well as a broad range of traditional and state-of-the-art anomaly detection algorithms. The benefit of OAB for the research community has been demonstrated by reproducing and extending existing benchmarks to new algorithms with very low effort allowing researchers to focus on the actual algorithm research.

Index Terms—Unsupervised and Semisupervised Anomaly Detection, Reproducibility, Benchmark, Evaluation, Datasets

I. INTRODUCTION

The increasing digitalization of processes in various domains like e.g. industrial automation, healthcare, mobility and others causes a large-scale volume of structured and unstructured data accommodating valuable knowledge and potential for process optimizations. One of the most interesting and similarly most domain beneficial kinds of knowledge discovery is the detection of abnormal patterns, also known as anomalies or outliers. "An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism." [1] The research field of anomaly detection demonstrates its relevance by addressing many applications like e.g. fraud detection, network security

issues, quality monitoring, etc. and thus it is not unusual that still numerous unsupervised and semisupervised anomaly detection algorithms [2], [3] get continuously developed and published. This advantageous situation of having such a versatile range of algorithms available becomes challenging every time when the question arises which algorithm performs better. There are still publications with only sparse or even no possibility for well reproducible results [4], missing an accurate description of hyperparameters, preprocessing steps, accessible data sets, applied sampling strategies, state of the art algorithms or meaningful standard evaluation metrics. Furthermore, in numerous anomaly detection benchmarks and papers (e.g. [5]–[9], etc.) anomaly data sets are synthetically generated by downsampling individual classes of actual classification data sets as anomalies instead of using real-world anomaly data sets containing semantically meaningful anomalies such as e.g. aerospace hardware parts on martian surface images [10]. Existing anomaly benchmarks [5], [6], [11] address all these requirements with different focus and scope but when researchers evaluate new algorithms they often cannot reproduce the results of competing algorithms. So they tend to conduct their experiments still individually instead of using existing benchmarks leading to not actually comparable results which makes truly well performing approaches hard to identify in the research field. Especially for anomaly detection algorithms developed for image and tabular data there is so far no standardized benchmark simultaneously ensuring (1) reproducibility, (2) comparability and (3) low-effort extensibility to new algorithms and data sets.

Thus we introduce with this work-in-progress the approach

of OAB - An **O**pen **A**nomaly **B**enchmark framework for image and tabular data sets supporting benchmarks for unsupervised and semisupervised anomaly detection algorithms and data sets. OAB covers a broad range of traditional and deep anomaly detection algorithms and provides next to most common anomaly benchmark data sets also real-world anomaly data sets. Moreover, OAB generalizes the task of un- and semisupervised anomaly benchmarks and sets an equally strong focus to the reproducibility and comparability of its results in order to support the requirements of the *Machine Learning Reproducibility Checklist* [4]. Different to other comparable benchmarks the OAB framework allows low-effort extensions to new algorithms and data sets enabling researchers to keep the focus on the actual algorithm research instead of trying to reproduce results and conducting their own individual experiments. The main contributions of this work can be summarized as follows:

- Simple reproducibility, comparability and extensibility for anomaly benchmark results when adding new algorithms or new data sets.
- OAB supports unsupervised and semisupervised benchmarks for image and tabular data sets by covering related traditional and deep anomaly detection algorithms.
- OAB offers reproducible sampling and contamination strategies and provides commonly used anomaly benchmark data sets as well as various real-world anomaly data sets.
- A well documented open Python library OAB with hands-on examples enabling the research community for its practical use.

The remainder is structured as follows. Section II describes related anomaly benchmarks. In Section III, we describe our framework for an Open Anomaly Benchmark and evaluate its performance w.r.t. the stated contributions in Section IV. Section V concludes the paper and proposes further ideas for future work.

II. RELATED WORK

In one of the first benchmarks for outlier detection the authors Campos et al. [5] present a study of unsupervised anomaly detection, in which an extensive collection of 23 data sets gets featured primarily taken from the UCI repository [12]. These data sets are all tabular, considering that ALOI is originally an image data set from which only premade image features are used, and are mostly rather low-dimensional and small in respect to the amount of samples. Considering the problem that originally most UCI data sets are intended for classification and sometimes clustering, Campos et al. especially highlight a group of data sets that are semantically meaningful for outlier detection in comparison to the common procedure of downsampling just any classification data set. These semantically meaningful data sets include medical data, i.e. normal and sick patients, mail and webdata, i.e. filtering out spam and ads, stamps (forged or real), and trees (normal or diseased). The benchmark only investigates k -nearest neighbor based outlier detection algorithms, since these were popular

at the time and are often still used as baseline models in current research (often represented by the Local Outlier Factor (LOF) algorithm). An advantage of this restriction is that all of these methods mainly depend on the size of the neighborhood that got analyzed at the beginning by the hyperparameter k . These methods are therefore easily compared in respect to this parameter. Campos et al. chose common evaluation measures for outlier detection, e.g. the Area under the Curve of the Receiver-Operating Characteristic (AUC ROC or ROC AUC) and Average Precision (AP), but highlight also a version of Precision-at- n ($P@n$), which was adjusted for chance (Adjusted $P@n$) similarly to AP (Adjusted $AP@n$). In order to prepare the data sets, different preprocessing steps are discussed in detail and listed for each data set. Campos et al. utilize downsampling for classification data sets and in case of duplicate samples a data set with and without duplication removal is provided. Categorical attributes get transformed into numerical data using either 1-of- n or Inverse Document Frequency (IDF). Furthermore, normalization gets considered as well as missing values if needed. Campos et al. published their implementations¹ by a collection of Python- and R-Scripts together with the used version of the Java-based KDD-Library ELKI [13].

The benchmark of Emmott et al. [6] also focuses like Campos et al. on unsupervised anomaly detection for tabular data. Therefore, 19 data sets are retrieved from the UCI repository [12]. In total 25685 child anomaly detection data sets are then sampled from these "mothersets" with variation in four dimensions. (1) Point difficulty measures how hard it is to distinguish normal points from anomalies, where the difficulty of an individual point is computed using Kernel Logistic Regression. (2) Semantic variation assesses how different the anomalies are from each other considering their clusteredness. (3) Contamination rate indicates what percentage of sampled data points are anomalous and (4) feature relevance/irrelevance adds additional irrelevant features to test how well an algorithm can handle these. Categorical features get discarded in advance from the mothersets, and all remaining features get normalized. Additionally, labels "normal" and "anomalous" are assigned to the data points.

A variety of 8 algorithms are tested on the resulting child data sets, including density-based algorithms like Ensemble Gaussian Mixture Model, model-based algorithms like One-Class Support Vector Machine, nearest-neighbors-based algorithms like LOF and projection-based ones like Isolation Forest. For the evaluation, the metrics ROC AUC and AP are used and their significance, and thus also the benchmark itself, gets evaluated by an additional hypothesis test.

All benchmarks are implemented in R and available² for download. As pointed out in the project's README, the documentation is unfortunately sparse. This published code allows to reproduce data preprocessing and sampling, the experiments and their results are not included however.

¹<https://www.dbs.ifi.lmu.de/research/outlier-evaluation/DAMI>

²<http://ir.library.oregonstate.edu/xmlui/handle/1957/59114>

One of the most recent comprehensive benchmarks for anomaly detection has been published by Domingues et al. [11]. Like the previously introduced benchmarks the focus is also set to unsupervised anomaly detection on tabular data. The collection of 15 data sets in total is composed out of 12 public data sets from the UCI [12] and OpenML [14] repositories and out of 3 proprietary data sets from the traveling industry domain. In order to benchmark for production relevant models by unseen data Domingues et al. apply train-test-splits in the unsupervised setting with an equal proportion of outliers in the train and test set. In extensive benchmarks 14 anomaly detection algorithms from different groups like probabilistic-, neighbourhood-, isolation-, SVM- or neural-network-based algorithm groups are evaluated in detail. The performance of these algorithms gets evaluated on test set samples by the metrics ROC AUC and Area Under the Precision Recall Curve (AUPRC). Furthermore, Domingues et al. conduct algorithm complexity analyses related to the time for training and prediction, memory usage and noise stability in dependence to the total number of samples and features based on synthetically generated data sets. In the preprocessing steps of the benchmark the data sets get normalized and the categorical data is one-hot-encoded allowing the algorithms to utilize the complete information from the data sets. Furthermore supporting reproducibility, the hardware setup, hyperparameters as well as the implementation languages like R, Python and Matlab are listed for each algorithm, but the benchmark scripts itself are not published avoiding a simple extension to new algorithms and data sets.

Besides the aforementioned well known approaches there are also further related benchmarks in the unsupervised setting like Goldstein et al. [15] which focuses on traditional algorithms and Lu et al. [16] evaluating dependency-based anomaly detection algorithms. Although each of them demonstrates various strengths, none of them allows the evaluation of most recent anomaly detection approaches in the un- and semisupervised setting on tabular as well as on image data sets and ensures a simple reproducibility, comparability and extensibility when new algorithms or new data sets should be added.

III. OPEN ANOMALY BENCHMARK (OAB)

This section describes the components of the Open Anomaly Benchmark framework covering requirements for image and tabular data sets, a selection of anomaly detection algorithm groups, most suitable evaluation metrics and methods ensuring reproducibility.

A. Benchmark Data Sets for Anomaly Detection

Ideally, actual anomaly data sets are used to benchmark anomaly detection algorithms, i.e., data sets with a semantic notion of *normal* and *anomalous* data points. However, real anomaly data sets are rare, especially in image detection, and might not have the characteristics authors are interested in when evaluating their new algorithm, e.g., with regards to number of observations or dimensionality. To remedy

this, classification and regression data sets are frequently transformed into anomaly detection data sets [6], [11]. In classification data sets, this assumes that a downsampled generating process of some class resembles an anomaly-generating process, which is not necessarily the case. However, as classification and regression data sets are frequently used to assess the performance of an anomaly detection algorithm, they are also provided by OAB and included in this paper. Note that OAB is not limited to working with the data sets provided by it, instead, also other data sets can easily be loaded making benchmarks with OAB simply extensible to own data sets.

B. Benchmark Algorithms for Anomaly Detection

This paper measures the performance of a variety of anomaly detection algorithms. OAB is in no way limited to working with these algorithms - it provides sampled data and an evaluation procedure for own algorithms to allow new approaches to be tested and replicated.

There are a variety of anomaly detection algorithms, distinguishing themselves in their assumptions about how normal data and anomalous data are different. We distinguish between traditional unsupervised, traditional semisupervised, and deep anomaly detection algorithms.

1) *Unsupervised algorithms*: The unsupervised algorithms receive a test data set as input and calculate anomaly scores of these test data points.

Nearest neighbour-based approaches (c.f. [6], [17]) assume that compared in its nearest neighbors, normal and anomalous data points behave differently in a specific way.

Isolation-based approaches aim to isolate anomalies and compute an anomaly score based on how difficult it is to isolate a data point [18].

Reconstruction-based approaches first reduce the dimensionality of a data point and then try to reconstruct it from this reduced-dimensional representation.

2) *Semisupervised algorithms*: The semisupervised algorithms are in contrast to unsupervised algorithms first trained on a clean training set. In the inference phase, they calculate an anomaly score for test points.

One-class approaches aim to learn a boundary around the normal training data points and assign anomaly scores based on this boundary.

Isolation-based approaches can also be used in the semisupervised setting. The rules to isolate data points are learned from the training set, and applied to data points from the test set [18].

Reconstruction-based approaches can be used similarly to isolation-based approaches, i.e., the dimensionality reduction and increase procedures are learned from the training data and applied to the test data.

3) *Deep learning algorithms*: The deep learning algorithms exist for both unsupervised and semisupervised anomaly detection and can be differentiated based on how directly the

deep part of the model is connected to anomaly detection. Among others, Pang et al. [2] identify the following two classes of algorithms:

Deep learning for feature extraction means that final layers for classification do not exist. The layers of such a neural network are only used to extract features which are fed into a traditional model.

Learning feature representations of normality comprise algorithms that are trained on some surrogate learning task that is not directly aiming for anomaly detection. Based on this surrogate task, a heuristic is used to calculate anomaly scores.

A core strength of OAB is that it is not limited to these algorithms. It allows easily testing new algorithms on the data sets provided and own data sets ensuring extensibility.

C. Evaluation Metrics

As the evaluation metrics of an anomaly benchmark framework need to be meaningful for the related task of anomaly detection several requirements need to be fulfilled. In the unsupervised setting there are usually no external labels available for evaluation. Thus, Marques et al. introduced IREOS [19] as an internal evaluation metric for the unsupervised setting. However, since the work of Aggarwal [20] states that internal validity measures are only rarely used for outlier detection and that it is more reasonable to use external validity measures in this field, OAB follows this recommendation and selects evaluation measures requiring the availability of external labels. Since the expressiveness of an evaluation metric is dependent on the used data set as well as on the task which an anomaly detection algorithm should perform there is not only one metric which should be taken into consideration. Hence the OAB framework supports the following set of evaluation metrics.

1) *ROC AUC*: The Receiver Operating Characteristic (ROC) [21] is represented by a curve in a chart defined by the true positive rate (TPR) of correctly detected outliers on the y-axis and by the false positive rate (FPR) of incorrectly detected outliers on the x-axis. This usually monotonously increasing ROC curve covers the full range of possible outlier decision thresholds τ with $\tau \in [0; 1]$. The Area Under that Curve (AUC), allows to represent this dependency by a single value between 0 and 1 describing the performance of the evaluated anomaly detection algorithm as ROC AUC. The maximum is a ROC AUC of 1.0 having a TPR of 1.0 and a FPR 0.0 from the beginning. The normalization of the TPR and FPR with the number of TP and FP respectively ensures ROC AUC still to be an expressive evaluation metric even when the ratio between normal and abnormal samples is imbalanced as it is usually the case for anomaly detection data sets. [5] In case the primary objective of the given anomaly detection task is to avoid false positives (false alarms) the ROC AUC would be a suitable evaluation measure.

2) *AUPRC*: As the name of the Precision-Recall Curve reveals it is defined by the Precision on the y-axis and the Recall on the x-axis for any possible decision threshold τ .

As for ROC AUC the Area Under the Precision-Recall Curve (AUPRC) describes the performance of an anomaly detection algorithm by a single value between 0 and 1 having the same maximum, but differently to ROC AUC the AUPRC is not necessarily monotonously increasing. If an anomaly detection task has the goal to avoid false positives (false alarms) as well as false negatives (missed alerts) then the AUPRC would be a reasonable evaluation metric.

3) *P@n*: The evaluation metric Precision@n [22] is conceived for benchmarking anomaly detection algorithms which focus on the evaluation of just the top n (ranked) outlier scores. This can be reasonable especially for large data sets with a relative high percentage of anomalies or in cases in which not more than n samples can or should be handled. Since the definition of n is crucial for meaningful P@n-results the availability of external anomaly labels is required. One possibility is to set n according to the total number of known anomalies in the data set. Since the precision is calculated with this metric, P@n can be used for anomaly detection tasks having their focus on the avoidance of false alarms.

4) *AdjP@n*: The Adjusted-Precision@n [5] addresses benchmark settings in which the outlier scores and thus also the percentage of total outliers may vary. Therefore, the previously introduced P@n metric gets adjusted for chance by aligning different outlier scores by their expected outlier score value.

5) *AP*: The Average Precision [23] evaluation metric is a further possibility to calculate the AUPRC. There exist several variants but since OAB relies on the `average_precision_score` of `sklearn` the weighted mean of precisions at each threshold is used. Equally to AUPRC, if an anomaly detection task needs to avoid false positives (false alarms) as well as false negatives (missed alerts) then the AP would be a suitable choice.

6) *AdjAP*: The Adjusted-Average-Precision [5] considers similarly to AdjP@n benchmark settings with varying outlier scores for an adjustment by chance, whereas the metric is calculated analogously.

7) *Friedman and Nemenyi tests*: The Friedman test [24] can be used to investigate whether or not all algorithms perform equally with regards to a metric, while the Nemenyi test [25] allows for a significance analysis of pair-wise performance comparisons.

D. Reproducibility

The core strengths of the OAB framework are reproducibility and comparability of its benchmarks. Therefore, OAB divides the task of anomaly benchmarking into a set of generic subtasks, which can as part of a benchmark recipe easily be replicated and generically applied to unsupervised and semisupervised anomaly benchmarks. These subtasks are represented by a variety of benchmarking steps introduced in this section. These benchmarking steps are closely related to the process steps for knowledge discovery and data mining (KDD) introduced by Fayyad et al. [26] (cf. Fig. 1a)). Differently to [26] the step of Data Mining is in OAB further divided into the

steps Sampling, which also represents the splitting into train and test sets, and Anomaly Detection Algorithms, leading to a Performance Overview after Evaluation. Considering Fig. 1b), OAB provides reproducibility across the benchmarking steps, noting that for steps with grey background no user-related actions are required and steps with white parts allow the user to increase or decrease the Selection of data sets and algorithms.

1) *Sampling Strategy*: Similar to algorithms, sampling strategies also need to be differentiated between unsupervised and semisupervised. With the sampling strategies outlined below, we aim to achieve two goals in both settings: (1) As many data points as possible are included to ensure that the result is not achieved by chance, and (2) to get a more robust estimate, we sample multiple times from each data set where each sample is different from the others.

In the unsupervised setting, the sampling should return a set of data points and their respective labels to allow for evaluation. Among the set of data points, the number of anomalies is expected to be comparatively low, as normal data points dominate the set of data points. The main sampling parameters are the number of samples s_u and the contamination rate c .

Target contamination rate. The target contamination rate, i.e., the proportion of anomalous data points among all data points, is set to $c = 0.05$ in our experiments, but a different target contamination rate can be set in OAB. The target contamination rate c is restricted to $c \in (0, 1)$.

Number of samples. The number of samples depends on the data set. In some data sets, the actual contamination rate is larger than the target contamination rate c , whereas in others, it is lower. Therefore, a procedure needs to be specified that calculates how many data points are sampled to ensure a contamination rate of 0.05. Emmott et al. [6] either subsample the normal data points if the original data's contamination rate is lower than the target contamination rate or subsample the anomalous data points in the other case. Our approach is similar to this and extends it in both providing a formalization and ensuring variance across samples. First, we calculate the maximum sample size restriction for normal and anomalous points s_n^{\max} and s_a^{\max} respectively. We denote the number of normal points in a data set by n_n and the number of anomalies by n_a .

$$s_n^{\max} = \left\lfloor \frac{n_n}{1-c} \right\rfloor \quad (1)$$

$$s_a^{\max} = \left\lfloor \frac{n_a}{c} \right\rfloor \quad (2)$$

If a data set consists of $n_n = 950$ normal data points and the target contamination rate is $c = 0.05$, the sample size can at maximum be $950/(1-0.05) = 1000$. If it is larger than this, a contamination rate of 0.05 is not possible without duplicating normal data points. This is calculated with Equation 1. The same train of thought can be applied to anomalous data points, leading to Equation 2. The tighter condition, i.e., the minimum of both values, is the maximum sample size. To ensure variability both among normal and anomalous labels,

this maximum sample size is scaled with a factor $f \in (0, 1)$ which we set to 0.9 to arrive at the actual sampling size in the unsupervised case s_u :

$$s_u = \left\lfloor f * \min \left\{ \frac{n_a}{c}, \frac{n_n}{1-c} \right\} \right\rfloor \quad (3)$$

Number of sampling steps and random seed. Finally, we sample a total of 10 times from a data set, and compute the average and standard deviation for each metric across these samples. For the first sample, the random seed is set to 42, and increased by 1 in each subsequent sampling step.

TABLE I
PARAMETERS USED WHEN SAMPLING IN THE UNSUPERVISED SETTING.

Target contamination rate c	0.05
Downscaling factor f	0.9
Sampling size s_u	$\left\lfloor f * \min \left\{ \frac{n_a}{c}, \frac{n_n}{1-c} \right\} \right\rfloor$
Number of sampling steps	10
Initial random seed	42

These sampling parameters are summarised in Table I. OAB allows for sampling either by specifying the number of data points to sample and the contamination rate or by specifying the contamination rate and a scaling factor. In the latter case, the sampling size is automatically computed using Equation 3.

In the semisupervised setting, a training set as well as a test set and the corresponding labels for the test set are provided by the sampling procedure. The training set is usually clean, i.e., it consists only of normal data points. In the test set, the contamination rate does not have to be as small as in the semisupervised case, as the model is already trained when assessing its performance on the test set and the data points from the test set do not affect the anomaly scores of each other. For the toothbrush data set of MVTEC AD for example, the test set consists of 11 normal points and 29 anomalies [27], [28]. The parameters for sampling are therefore not specified by the number of samples and the contamination rate, but instead by the percentage *train* of normal data points used for training and the maximum contamination rate c_{test}^{\max} in the test set.

Percentage of normal data points used for training. The parameter *train* specifies the percentage of normal data points which are used for training. It is constrained to $train \in (0.5, 1)$ as there should be more observations in the training set than in the test set, and $1 - train$ is the part of the normal data points used for testing. *train* needs to balance two considerations. On the one hand, a larger training set allows the algorithm to see more normal data points which in turn allows it to better learn what normal data points look like. On the other hand, the training set should not be too large, as in this case the number of normal data points in the test set can become very small. If this is the case and the test set contains mainly anomalous points, it becomes more difficult to assess an algorithm's strength in distinguishing normal from anomalous points. A training set size of $train = 0.7$ of the normal points is therefore chosen.

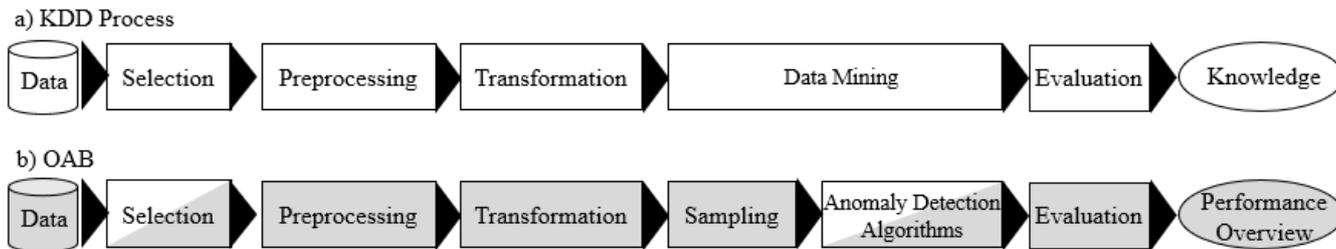


Fig. 1. Comparison of a) KDD process steps [26] and b) reproducible benchmarking steps of OAB (grey) and user-related steps (white).

Maximum contamination rate. The test set consists of both normal and anomalous data points. All remaining normal data points are used in the test set, and theoretically, all anomalous data points can be used as well. Note however that in some data sets, especially original classification data sets with multiple classes, this can lead to a large contamination rate in excess of 0.9. If the ratio of normal points in the test set is very small, the performance metric loses robustness. For this reason, the contamination rate in the test set is restricted by $c_{test}^{max} \in (0, 1)$, which is set to 0.5 in this benchmark.

TABLE II

PARAMETERS USED WHEN SAMPLING IN THE SEMISUPERVISED SETTING.

$train$	0.7
c_{test}^{max}	0.5
Number of samples	10
Initial random seed	42

Apart from the parameters specified here, the random seed and number of samples are the same as in the unsupervised setting. The parameters are summed up in Table II.

Note that in contrast to unsupervised sampling, semisupervised sampling does not need a scaling factor as the variability is ensured by splitting the normal data into different training and test sets in each sampling. In addition to the sampling described above, OAB also supports sampling using explicit training and test set sizes and allows for a contamination in the training set, which can be of use to assess an algorithm’s robustness towards anomalies in the training set. In this benchmark however, the sampling above is used.

If data sets already have a native train-test split like MVTEC AD [27], [28], this split can also be used for sampling using OAB. This native split is also used in the experiments section of this paper.

2) *Preprocessing, Transformation and Parameters:* To ensure reproducibility, describing which data set is used and where it was downloaded from does not suffice. If preprocessing and transformation steps are applied, they have to be well-documented. In addition to that, also the training- and hyperparameters need to be tracked. Therefore, this section describes these steps in detail.

Regression data sets. While classification data sets can easily be transformed into anomaly detection data sets by specifying which labels are considered to be normal and which

anomalous, regression data sets need a different approach: The inter-quartile range (IQR) is calculated and instances x are considered as outliers in case of exceeding the range of $[(Q1 - d * IQR); (Q3 + d * IQR)]$ where d is a data set specific factor (mostly $d=1.5$).

Missing values. Algorithms considered here are not capable of working with data points with missing features. When missing values do occur, we follow Campos et al. [5] and delete an attribute if 10% or more of the instances do not have a value for this attribute. Otherwise only the related instances get removed.

Duplicate data points. In line with the preprocessing performed by Campos et al. [5], duplicate data points are removed.

Categorical features. While most features are numeric, some categorical features also exist. Campos et al. [5] propose two techniques to deal with categorical features, namely one-hot encoding and using the inverse document frequency of the attribute value as encoding. Emmott et al. [6] ignore categorical features. In this benchmark, categorical features are one-hot encoded.

Parameters. The OAB recipe also tracks training- and hyperparameters for reproducible algorithm runs. These are e.g. the number of epochs, learning rate, batch size, layer sizes, etc.

Normalization. Campos et al. [5] propose to use two data sets: One with attribute-wise linear normalization to values between 0 and 1 (including) and a second one without normalization. Domingues et al. [11] on the other hand standardize their features, i.e., scale them to mean 0 and unit standard deviation. For tabular data, attribute-wise scaling with `RobustScaler` of `sklearn` on all attributes is used. For image data, the typical machine learning scaling to values between 0 and 1 is used, i.e., all pixel values are multiplied by $1/255$.

All aspects described above are part of OAB and thus reproducible for each experiment.

IV. EXPERIMENTAL SETUP AND RESULTS

In this section, the performance of the OAB framework gets investigated. Therefore, a selection of tabular and image data sets is loaded, the data sets are sampled and the selected anomaly detection algorithms get benchmarked on these samples. The results can easily be replicated using the Google-Colabs from our github.

A. Data sets

As described in Section III-A, OAB supports both real anomaly data sets as well classification and regression data sets transformed into anomaly data sets. The data sets used to obtain the benchmark results are presented in this section. They are naturally divided into tabular and image data sets. Real anomaly data sets, i.e., data sets that contain semantically meaningful anomalies, are marked with *. In addition to these data sets, a variety of other data sets is also built in to OAB.³ The following tabular data sets are used in this study. A brief description of their characteristics can be found in Table III.⁴

Spambase* [12]. Emails divided into non-spam (normal) and spam (anomalous)

Wilt* [12]. Segments of images divided into other land area (normal) and diseased trees (anomalous)

NASA* [29]. NASA software for receiving and processing ground data divided into non-failing (normal) and with defects (anomalous)

Annthyroid* [30]. Medical data, divided into healthy (normal) and two kinds of hypothyroidism (anomalous)

Page-blocks [12]. Blocks of a page layout divided into text blocks (normal) and non-text blocks (anomalous)

Ionosphere [30]. Radar returns from the ionosphere, classified into good (normal) or bad (anomalous)

Boston [31]. Regression data set on housing data and prices for areas in Boston, transformed as described above

Image data sets consist of either colored or black-and-white images. Their characteristics are shown in Table IV.

MVTec AD* [27]. MVTec AD consists of a set of real-world data sets with object and texture images. From this collection, the data sets transistor, screw, pill, carpet and hazelnut (hazel) are used.

MNIST_0 (mnist) [32]. Hand-written digits classification data set, with instances for each digit from 0 to 9. Here, we transform it into an anomaly detection data set by setting the label 0 as normal label and all others as anomalous, which also defines the suffix of the data set name.⁵

CIFAR-10_0 (cifar10) [33]. Classification data set with 10 classes of images. Here, we set label 0 as normal label and all others as anomalous.

B. Algorithms

In Section III-B, a variety of different categories of algorithms was laid out. At least one algorithm from each category is presented here and used in the experiments. Unless noted otherwise, PyOD's [34] implementation is used. For each setting, we provide a table indicating which algorithms are used in this setting and which hyperparameters they are used with to ensure reproducibility. If hyperparameters are not mentioned, the defaults of PyOD v0.9.3 are used.

³A list of all built-in data sets is provided in the supplementary github.

⁴For details please refer to the supplementary material on github.

⁵OAB also supports iterating through all labels as normal label to obtain a single score for all anomaly data sets composable from MNIST.

TABLE III

TABULAR DATA SETS. n_{NORMAL} IS THE NUMBER OF NORMAL DATA POINTS, AND n_{ANOMALY} THE NUMBER OF ANOMALIES IN THE DATA SET. FEATURES INDICATES THE NUMBER OF FEATURES (OR DIMENSIONALITY) OF EACH DATA POINT.

Name	n_{normal}	n_{anomaly}	Features
spambase	2528	1679 (39.91%)	57
wilt	4562	257 (5.33%)	5
NASA	877	315 (26.43%)	21
annthyroid	6528	534 (7.56%)	6
page-blocks	4883	510 (9.46%)	10
ionosphere	225	125 (35.71%)	33
boston	475	31 (6.13%)	13

TABLE IV

IMAGE DATA SETS. THE FINAL FIVE DATA SETS ARE FROM MVTEC AD.

Name	n_{normal}	n_{anomaly}	Features
MNIST_0	6903	63097 (90.15%)	(28, 28)
CIFAR10_0	6000	54000 (90%)	(32, 32, 3)
transistor	273	40 (12.78%)	(256, 256, 3)
screw	361	119 (24.79%)	(256, 256)
pill	293	141 (32.49%)	(256, 256, 3)
carpet	308	89 (22.41%)	(256, 256, 3)
hazelnut	431	70 (13.97%)	(256, 256, 3)

For tabular data, Table V covers the unsupervised setting and Table VI covers the semisupervised setting. For image data, please refer to Tables VII and VIII respectively. In the following the core concepts of each benchmarked anomaly detection algorithm gets briefly described.

Nearest neighbor-based approaches: kNN. The vanilla k-nearest neighbors (kNN) algorithm uses the distance to the k-th neighbour of a data point as anomaly score. The underlying assumption is that normal data points can be found in dense neighborhoods, which is not the case for anomalous points [17]. The choice for k is inspired by the choice of Domingues et al. [11] for similar algorithms.

Nearest neighbor-based approaches: LOF. As the name Local Outlier Factor [35] indicates, the neighbourhood of a point is investigated to judge if a data point is anomalous instead of observing all data points globally. If a data point is in a less dense neighborhood than its neighbors, it is assumed to have a higher likeliness of being an anomaly. The choice for k is taken from the choice of Domingues et al. [11], but a lower bound is chosen as some data sets have few observations.

Nearest neighbor-based approaches: ABOD. Angle-Based outlier Detection was proposed by Kriegel et al. [36] and assumes that the variation in angles from a data point to other data points is larger for normal data points, as they can be found towards the center of a dense cluster, and smaller for anomalies. As for kNN and LOF, the hyperparameter k is inspired by the choice of Domingues et al. [11].

Isolation-based approaches: Isolation Forest. Isolation Forests [18] consist of isolation trees which are trees built at random. A data point traverses the trees and it is assumed that anomalies are isolated, i.e., reach a leaf node, faster than normal data points.

One-class approaches: OCSVM. One-Class Support Vector Machines [37] map data points into a feature space and learn a hyperplane that separates the normal training data points from the origin as tightly as possible. Test data points are assigned anomaly scores based on their position relative to the hyperplane in the feature space.

Reconstruction-based approaches: PCA. Principal component analysis [38], [39] is a linear model that can be used to reduce dimensionality. From this reduced dimensional-representation, the original representation is recomputed using the eigenvectors. The respective mappings can either be learned on a clean training set (semisupervised) or on a test set in which normal data points dominate (unsupervised). The reconstruction error is transformed into an anomaly score.

Deep learning for feature extraction: AE + Traditional algorithm. Autoencoders (AE) [40] can be used as a means to reduce feature dimensionality. The bottleneck representation can be fed into a traditional algorithm. For image data, convolutional AEs (CAEs) are usually used. CAEs as well as the combination of two algorithms are self-implemented.

Learning feature representations of normality: AE. AEs can be used to reduce dimensionality, but analogous to PCA, they can also be used as anomaly detection algorithm in both the unsupervised and semisupervised setting. As in PCA, the anomaly scores are calculated based on the reconstruction error of a data point. Alternatives are Variational AEs (VAEs) [41] and CAEs.

TABLE V
HYPERPARAMETERS FOR ALGORITHMS USED IN UNSUPERVISED TABULAR DATA ANOMALY DETECTION. n IS THE NUMBER OF DATA POINTS IN A SAMPLE.

kNN	n_neighbors	$\max(n * 0.05, 10)$
LOF	n_neighbors	$\max(n * 0.1, 10)$
ABOD	n_neighbors	$\max(n * 0.01, 10)$
IForest	random_state	42
AE	random_state	42
	hidden_layers	[6, 3, 3, 6]
AE+LOF	AE: random_state	42
	AE: hidden_layers	[6, 3, 3, 6]
	LOF: n_neighbors	$\max(n * 0.1, 10)$

TABLE VI
HYPERPARAMETERS FOR ALGORITHMS USED IN SEMISUPERVISED TABULAR DATA ANOMALY DETECTION.

OCSVM	–	–
IForest	random_state	42
PCA	n_components	0.9
	pca_solver	'full'
AE	random_state	42
	hidden_layers	[6, 3, 3, 6]
VAE	random_state	42
	encoder_neurons	[6, 3]
	decoder_neurons	[3, 6]

TABLE VII
HYPERPARAMETERS FOR ALGORITHMS USED IN UNSUPERVISED IMAGE DATA ANOMALY DETECTION. n IS THE NUMBER OF DATA POINTS IN A SAMPLE. THE CAE PARAMETERS ARE ALWAYS THE SAME AS THOSE IN THE LAST ROW.

CAE+kNN	kNN: n_neighbors	$\max(n * 0.05, 10)$
CAE+LOF	LOF: n_neighbors	$\max(n * 0.1, 10)$
CAE+ABOD	ABOD: n_neighbors	$\max(n * 0.01, 10)$
CAE+IForest	IForest: random_state	42
CAE	random_state	42
	epochs	50
	latent_dim	100

TABLE VIII
HYPERPARAMETERS FOR ALGORITHMS USED IN SEMISUPERVISED IMAGE DATA ANOMALY DETECTION. THE CAE PARAMETERS ARE ALWAYS THE SAME AS THOSE IN THE LAST ROW.

CAE+OCSVM	–	–
CAE+IForest	IForest: random_state	42
CAE	random_state	42
	epochs	50
	latent_dim	100

C. Results

We ran the algorithms mentioned above on the data sets presented in Section IV-A. The sampling parameters are those described in Tables I and II in Section III-D1 and the MVTEC AD data sets were sampled with their native train-test split in the semisupervised setting. Results for tabular data sets can be found in Tables IX and X for unsupervised and semisupervised anomaly detection. The results for image data sets can be found in Tables XII and XI respectively.

D. Reproduce results from other benchmarks

To further validate OAB, we aimed at reproducing results from other papers. As the preprocessing is in large parts comparable to Campos et al. [5], we focussed on their results. Because they report results for each sampled data set individually and with varying hyperparameters, we were specifically interested in algorithms which had the same or very similar hyperparameters for each sampled data set. We found that this is the case for LOF with n_neighbors=100 on the PageBlocks data set⁶ with a contamination rate of 0.05 in its unscaled variant without duplicates. We further reproduced their sampling procedure by setting the number of sampled points to 5139. As shown in the corresponding Google-Colab accompanying this paper on github, we were able to reproduce their results. The ROC AUC score was the same up to 3 digits after the decimal, and other scores matched up to 2 digits after the decimal.

V. CONCLUSION

In summary, we introduced in this paper OAB, an Open Anomaly Benchmark framework for unsupervised and semisupervised anomaly detection on image and tabular data sets.

⁶<https://www.dbs.ifi.lmu.de/research/outlier-evaluation/DAMI/semantic/PageBlocks/>

TABLE IX
TABULAR DATA IN UNSUPERVISED SETTING, ROC AUC WITH STANDARD DEVIATIONS.

	spambase	wilt	NASA	anthyroid	page-blocks	ionosphere	boston	Average
kNN	0.633±0.025	0.612±0.005	0.653±0.034	0.947±0.002	0.935±0.004	0.950±0.030	0.733±0.008	0.780
LOF	0.588±0.025	0.573±0.006	0.652±0.032	0.950±0.002	0.927±0.004	0.907±0.040	0.715±0.010	0.758
IForest	0.778±0.014	0.414±0.027	0.671±0.031	0.826±0.008	0.914±0.004	0.947±0.026	0.811±0.018	0.766
ABOD	0.721±0.016	0.699±0.005	0.698±0.029	0.927±0.005	0.954±0.004	0.970±0.022	0.632±0.042	0.800
AE	0.753±0.019	0.334±0.008	0.593±0.040	0.688±0.011	0.915±0.005	0.917±0.035	0.796±0.012	0.714
AELOF	0.504±0.005	0.500±0.000	0.506±0.008	0.503±0.004	0.524±0.032	0.503±0.013	0.470±0.004	0.501
Average	0.663	0.522	0.629	0.807	0.861	0.866	0.693	

TABLE X
TABULAR DATA IN SEMISUPERVISED SETTING, ROC AUC WITH STANDARD DEVIATIONS.

	spambase	wilt	NASA	anthyroid	page-blocks	ionosphere	boston	Average
OCSVM	0.658±0.009	0.463±0.008	0.618±0.014	0.953±0.002	0.944±0.003	0.902±0.021	0.738±0.023	0.754
IForest	0.821±0.011	0.452±0.026	0.663±0.013	0.904±0.011	0.929±0.004	0.923±0.027	0.838±0.014	0.790
PCA	0.810±0.007	0.327±0.040	0.587±0.015	0.792±0.013	0.929±0.004	0.903±0.024	0.812±0.017	0.737
AE	0.809±0.007	0.351±0.006	0.565±0.019	0.822±0.012	0.934±0.004	0.914±0.021	0.819±0.012	0.745
VAE	0.810±0.007	0.341±0.006	0.585±0.017	0.823±0.012	0.933±0.004	0.902±0.022	0.814±0.015	0.744
Average	0.782	0.387	0.604	0.859	0.934	0.909	0.804	

TABLE XI
IMAGE DATA IN SEMISUPERVISED SETTING, ROC AUC WITH STANDARD DEVIATIONS. NOTE THAT FOR MVTEC AD DATA SETS, THE ORIGINAL TRAIN-TEST SPLIT IS USED. THEREFORE, DATA IS NOT SAMPLED MULTIPLE TIMES AND THERE IS NO STANDARD DEVIATION.

	mnist	cifar10	transistor	screw	pill	carpet	hazelnut	Avg
CAEOCSVM	0.991±0.001	0.639±0.009	0.721	0.720	0.502	0.614	0.632	0.688
CAEIForest	0.980±0.003	0.641±0.014	0.718	0.269	0.524	0.519	0.661	0.616
CAE	0.943±0.005	0.714±0.008	0.675	0.001	0.509	0.492	0.841	0.597
Avg	0.971	0.665	0.705	0.330	0.512	0.542	0.711	

TABLE XII
IMAGE DATA IN UNSUPERVISED SETTING, ROC AUC SCORES.

	mnist	cifar10	transistor	screw	pill	carpet	hazel	Avg
CAEKNN	0.992	0.639	0.649	0.486	0.686	0.556	0.554	0.652
CAELOF	0.995	0.632	0.657	0.526	0.790	0.541	0.548	0.670
CAEABOD	0.935	0.657	0.686	0.567	0.676	0.569	0.671	0.680
CAEIForest	0.967	0.645	0.638	0.507	0.575	0.521	0.545	0.628
CAE	0.952	0.737	0.635	0.360	0.643	0.381	0.401	0.587
Avg	0.968	0.662	0.653	0.489	0.674	0.514	0.544	

With this work-in-progress we did neither aim to find the best performing anomaly detection algorithm nor replace well established benchmarking methods or try to cover each algorithm or data set ever benchmarked in this context. Instead, we demonstrated that OAB allows to standardize benchmarking related steps like preprocessing, sampling, train-test-splitting as well as the actual evaluation while simultaneously ensuring reproducibility, comparability and low-effort extensibility for new anomaly detection algorithms and data sets. Thereby, OAB sets the foundation towards an automatized fulfillment of the requirements of the *Machine Learning Reproducibility Checklist* [4].

Furthermore, the OAB framework provides a selection of semantically meaningful real-world anomaly data sets and covers different methods for preprocessing, sampling and contamination. Addressing the openness of the framework, OAB is open for further contributions by sharing the code

as open-source project⁷ but it is also open in relation to offering existing benchmarks in detail (data sets, algorithms, benchmark setups, etc.). The practical use of OAB is supported by well documented Google-Colabs shared in the github repository for hands-on experience.

Since a reasonable benchmark should not only consider the performance of an algorithm on a set of arbitrarily selected data sets for specific evaluation metrics, OAB would benefit from future work evaluating algorithms related to their dependency on the choice of normalization method and on the data set characteristics like already demonstrated by Kandanaarachchi et al. [42]. Moreover, as there are no external labels available in the actual unsupervised setting, the sampling of OAB could be further improved by sampling from different embedding clusters to ensure a point difficulty balance in case of downsampling or train-test-splitting.

ACKNOWLEDGMENT

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

⁷<https://github.com/ISDM-CAU-Kiel/oab>

REFERENCES

- [1] D. M. Hawkins, "Identification of outliers," in *Monographs on Applied Probability and Statistics*, 1980.
- [2] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, vol. 54, 2021.
- [3] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Muller, "A unifying review of deep and shallow anomaly detection," *Proceedings of the IEEE*, vol. 109, p. 756–795, 2021.
- [4] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Lariviere, A. Beygelzimer, F. d'Alche Buc, E. Fox, and H. Larochelle, "Improving reproducibility in machine learning research (a report from the neurips 2019 reproducibility program)," *Journal of Machine Learning Research*, vol. 22, pp. 1–20, 2021. [Online]. Available: <http://jmlr.org/papers/v22/20-303.html>
- [5] G. O. Campos, A. Zimek, J. Sander, R. J. Campello, B. Micenkova, E. Schubert, I. Assent, and M. E. Houle, "On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study," *Data Min. Knowl. Discov.*, vol. 30, p. 891–927, Jul. 2016.
- [6] A. Emmott, S. Das, T. G. Dietterich, A. Fern, and W.-K. Wong, "A meta-analysis of the anomaly detection problem," *arXiv: Artificial Intelligence*, 2015.
- [7] F. Keller, E. Muller, and K. Bohm, "Hics: High contrast subspaces for density-based outlier ranking," in *2012 IEEE 28th International Conference on Data Engineering*, 2012, pp. 1037–1048.
- [8] A. Zimek, M. Gaudet, R. J. G. B. Campello, and J. Sander, "Subsampling for efficient and effective unsupervised outlier detection ensembles," *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013.
- [9] J. Yang, N. Zhong, Y. Yao, and J. Wang, "Local peculiarity factor and its application in outlier detection," in *KDD*, 2008.
- [10] A. McEwen, "Views of msl hardware 12 days after landing," 2012. [Online]. Available: https://www.uahirise.org/ESP_028401_1755
- [11] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, "A comparative evaluation of outlier detection algorithms: Experiments and analyses," *Pattern Recognit.*, vol. 74, pp. 406–421, Sep. 2018.
- [12] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [13] E. Schubert and A. Zimek, "ELKI: A large open-source library for data analysis - ELKI release 0.7.5 "heidelberg"," *CoRR*, vol. abs/1902.03616, 2019.
- [14] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "Openml," *ACM SIGKDD Explorations Newsletter*, vol. 15, p. 49–60, 2014.
- [15] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PLoS ONE*, vol. 11, 2016.
- [16] S. Lu, L. Liu, J. Li, T. Le, and J. Liu, "Dependency-based anomaly detection: Framework, methods and benchmark," *ArXiv*, vol. abs/2011.06716, 2020.
- [17] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, pp. 15:1–15:58, 07 2009.
- [18] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Trans. Knowl. Discov. Data*, vol. 6, Mar. 2012.
- [19] H. Marques, R. J. G. B. Campello, A. Zimek, and J. Sander, "On the internal evaluation of unsupervised outlier detection," *Proceedings of the 27th International Conference on Scientific and Statistical Database Management*, 2015.
- [20] C. C. Aggarwal, *Outlier Analysis*, 2nd ed. Springer Publishing Company, Incorporated, 2016.
- [21] J. Hanley and B. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve," *Radiology*, vol. 143 1, pp. 29–36, 1982.
- [22] N. Craswell, *Precision at n*. Boston, MA: Springer US, 2009, pp. 2127–2128.
- [23] E. Zhang and Y. Zhang, *Average Precision*. Boston, MA: Springer US, 2009, pp. 192–193.
- [24] M. Friedman, "A Comparison of Alternative Tests of Significance for the Problem of m Rankings," *The Annals of Mathematical Statistics*, vol. 11, pp. 86–92, 1940.
- [25] P. Nemenyi, *Distribution-free Multiple Comparisons*. Princeton University, 1963.
- [26] U. M. Fayyad, G. Piatesky-Shapiro, and P. Smyth, "Knowledge discovery and data mining: Towards a unifying framework," in *KDD*, 1996.
- [27] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "Mvtec ad – a comprehensive real-world dataset for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [28] P. Bergmann, K. Batzner, M. Fauser, D. Sattlegger, and C. Steger, "The mvtec anomaly detection dataset: A comprehensive real-world dataset for unsupervised anomaly detection," *Int. J. Comput. Vis.*, vol. 129, pp. 1038–1059, 2021.
- [29] J. Sayyad Shirabad and T. Menzies, "The promise repository of software engineering databases," <https://www.openml.org/d/1067>, 2005.
- [30] S. Rayana, "Odds library," 2016. [Online]. Available: <http://odds.cs.stonybrook.edu>
- [31] D. Harrison and D. L. Rubinfeld, "Hedonic housing prices and the demand for clean air," *Journal of Environmental Economics and Management*, vol. 5, pp. 81–102, 1978.
- [32] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, pp. 141–142, 2012.
- [33] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009. [Online]. Available: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>
- [34] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, pp. 1–7, 2019.
- [35] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, p. 93–104, 2000.
- [36] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 444–452.
- [37] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating support of a high-dimensional distribution," *Neural Computation*, vol. 13, pp. 1443–1471, 07 2001.
- [38] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, pp. 559–572, 1901.
- [39] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, pp. 498–520, 1933.
- [40] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [41] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2014.
- [42] S. Kandanaarachchi, M. A. Muñoz, R. Hyndman, and K. Smith-Miles, "On normalization and algorithm selection for unsupervised outlier detection," *Data Mining and Knowledge Discovery*, vol. 34, pp. 309–354, 2019.

Chapter 8

AI4EO Hyperview: A SpectralNet3D and RNNPlus Approach for Sustainable Soil Parameter Estimation on Hyperspectral Image Data

This chapter consists of a preprint version of the following publication:

Claudius Zelenka, Andreas Lohrer, Mirjam Raffaella Bayer, and Peer Kröger. “AI4EO Hyperview: A SpectralNet3D and RNNPlus Approach for Sustainable Soil Parameter Estimation on Hyperspectral Image Data”. In: *2022 IEEE International Conference on Image Processing, ICIP 2022, Bordeaux, France, 16-19 October 2022*. IEEE, 2022, pp. 4263–4267. DOI: 10.1109/ICIP46576.2022.9897889. URL: <https://doi.org/10.1109/ICIP46576.2022.9897889>

- **Conception:** Lohrer (Lead), Zelenka (Lead)
- **Planning:** Lohrer (Lead), Zelenka (Support)
- **Execution:** Lohrer (Lead), Zelenka (Support), Bayer (Support)
- **Manuscript:** Zelenka (Lead), Lohrer (Lead), Bayer (Support), Kröger (Support)

AI4EO HYPERVIEW: A SPECTRALNET3D AND RNNPLUS APPROACH FOR SUSTAINABLE SOIL PARAMETER ESTIMATION ON HYPERSPECTRAL IMAGE DATA

Claudius Zelenka¹, Andreas Lohrer^{1,2*}, Mirjam Bayer^{1◦}, Peer Kröger¹

¹Kiel University, Kiel, Germany ²LMU Munich, Munich, Germany

ABSTRACT

The goal of the #Hyperview challenge is to use Hyperspectral Imaging (HSI) to predict the soil parameters potassium (K), phosphorus pentoxide (P_2O_5), magnesium (Mg) and the pH value. These are relevant parameters to determine the need of fertilization in agriculture. With this knowledge, fertilizers can be applied in a targeted way rather than in a prophylactic way which is the current procedure of choice.

In this context we introduce two different approaches to solve this regression task based on 3D CNNs with Huber loss regression (SpectralNet3D) and on 1D RNNs. Both methods show distinct advantages with a peak challenge metric score of 0.808 on provided validation data.

Index Terms— 3D Convolutional Neural Net (3DCNN), Recurrent Neural Network (RNN), Regression, Soil Parameter Prediction, Remote Sensing

1. INTRODUCTION

Using hyperspectral images (HSI) to predict soil parameters is part of the domain remote sensing in agriculture. When these parameters are obtained timely the fertilization can be spread more targeted and over fertilization can be reduced.

This challenge aims to train an estimation of the soil parameters potassium (K), phosphorus pentoxide (P_2O_5), magnesium (Mg) and the pH value for agricultural fields in Poland. HSI has been used for soil parameter predictions on tilled agricultural fields [1], subtropical coastal wetlands [2] and others. A further utilization of HSI is to predict moisture of coastal saline, which has been investigated in [3].

In the area of remote sensing a variety of approaches and adaptations like partial least-square regression (PLSR) [4] investigate in the regression problem of estimating soil parameters based on HSI data. In [5] the combination with a weighting method is proposed to overcome the difficulty of redundancy. Multiple linear regression (MLR) approaches have been shown suitable for soil parameter estimation [3] and

for soil mapping [6]. The use of Gaussian process regression (GPR) was tested in [7] in combination with wavelet transform for soil character prediction. In [8] multiple approaches for the prediction of six soil iron parameters are compared. It is shown that Support Vector Machine Regression (SVMR) outperformed PLS for most parameters. Other applications for remote sensing data are covered in the survey [9]. Separate approaches for classification and soil mapping using deep learning are presented in [10]. In [11] it is shown that CNN approaches outperform PLS significantly on the LUCAS soil database [12], predicting six soil properties in one model. Tying to these insights we experiment with two different approaches to estimate four soil parameters.

With this work in progress paper we introduce the following contributions:

- a RNNPlus based reflectance curve approach inspired by time-series regression which allows the prediction of soil parameters from very low spatial resolution images with low computational cost.
- SpectralNet3D, a 3DCNN regression approach for soil parameter prediction based on Huber loss and 3D convolutional feature learning from HSI cube wavelength ranges.

1.1. Challenge

The #Hyperview challenge "Seeing beyond the visible: Estimating soil parameters from hyperspectral images"¹ is hosted by the KP Labs together with the European Space Agency (ESA) and QZ Solutions. It is part of the IEEE International Conference on Image Processing (ICIP) Conference 2022 as a grand challenge². The goal of the challenge is to find the best solution for predicting soil parameters using HSI. The ground truth values have been collected using the current state of the art method for laboratory procedures. This is a time consuming process, since the soil samples have to be taken from all fields of interest and then have to be analysed manually in laboratories.

With the winning code of the challenge, a nanosatellite gets prepared to deliver timely information for farmers about

* Author of SpectralNet3D and funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A.

◦ Further parts of this work have been done in the MARISPACE-X project funded by German Ministry of Economy (BMW). The authors of this work take full responsibility for its contents.

¹<https://platform.ai4eo.eu/seeing-beyond-the-visible>

²<https://2022.ieeeicip.org/challenges/>

the state of their land. This allows a more precise application and hence a reduced usage of fertilizers enabling a step towards improved and sustainable agriculture practices.

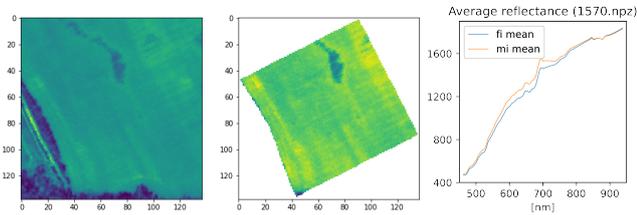


Fig. 1. Challenge data sample showing a raw hyperspectral image (left), its masked region of interest for soil parameter estimation (middle) and the 1D-average reflectance (right) of the full scene and the masked region of interest.

2. RELATED WORK

AI4EO Baseline Regressor In scope of the challenge introduced in section 1.1 a baseline³ was provided. The *Baseline Regressors* performance represents the starting point for the competition and acts hence as the denominator for the performance score metric introduced in section 4.2. This approach uses the training data to calculate the mean value for each target feature and returns them unchanged for all samples of a given test data set. As simple and intuitive this approach is from preprocessing and runtime perspective, it neglects the magnitudes of change between different hyperspectral bandwidths besides the information about spatial densities.

SpectralNET Similar to the waveletCNN[13] model the SpectralNET[14] approach enriches the usage of 2D CNNs with multiresolutional analysis by incorporating wavelet transformation into the architecture.

Instead of 2D convolutions on Haar wavelet transform[15] decompositions indicated as high and low pass filters the proposed SpectralNet3D approach (cf. section 3.2) utilizes 3D convolutions on masked HSI cube wavelength ranges with the goal of mapping higher spectral information densities. Differently to SpectralNET, which has the task of HSI based classification, SpectralNet3D is optimized for the regression task of soil parameter prediction.

M3D-DCNN He et al. [16] introduces a multi-scale 3DCNN combining 1D-, 2D- and 3D-feature extraction for spatial and spectral information in one architecture. Instead of HSI classification SpectralNet3D targets less complexity with pure 3D feature extraction for soil parameter regression.

³<https://github.com/AI4EO/kp-labs-seeing-beyond-visible-challenge>

3. MODELS

3.1. RNNPLUS

For every hyperspectral image an average reflectance curve can be generated by calculating the mean value over the entire patch for each wavelength. This is equivalent to using a 1-pixel camera or a non-imaging spectral light sensor. Such an approach has the advantage that high-resolution spatial information from sensors is not necessary and a much lower resolution could be used. Therefore the 150 wavelengths image stack is reduced to an 150 values 1D sequence.

An example can be seen in Figure 1. For this approach we make the assumption that we can predict the soil parameter only from the reflectance curve without texture information.

Because of the physical properties of soil we can expect a continuous reflectance spectrum which is further smoothed by any imperfect properties of the sensor. This strong correlation between adjacent wavelengths can easily be seen in Figure 1. Strong correlations between values mean that there is a interdependence between these values which should be used for optimal results. Hence, instead of applying general tabular machine learning (ML) algorithms, we seek an algorithm that exploits this interdependence.

This motivates applying a time series classification and regression algorithm, we use the 'RNNPlus' regression from [17]. It is a combination of a 1D-CNN feature extractor with an RNN-backbone which consists of 100 units in a hidden layer. This is followed by a fully connected linear layer head to adjust the output size to four parameters. For preprocessing we apply input-standardization with both the mean and standard deviation of the entire training set separately for each soil parameter. The network is trained over 20 epochs with a learning rate of 0.0003.

The smooth curve and high correlation between adjacent wavelength noted above imply a redundancy in the data, thus we reduce the sequences to 30 values by even sampling, which further lessens the computing requirements and cuts down the requirements on the hyper-spectral camera.

3.2. SPECTRALNET3D

This section introduces SpectralNet3D*, a neural network approach for regression optimizing 3D convolutional HSI feature learning for soil parameter prediction.

3.2.1. Architecture

The regression task of predicting soil parameters based on 3-dimensional HSI data requires a neural network architecture which is capable to extract spatial information as well as wavelength-related information from multiple bandwidths, and is able to map this information to the desired target soil parameter values. These parts for feature learning and prediction are covered by the proposed architecture in Table 1

which has been empirically determined. Starting with the input layer (L0) of 1x5x284x284 (Cx Dx Hx W) the HSI data cube represents the spatial information by an image height and width of 284 pixels, which is the maximum size of the masked size-varying, centered and padded HSI data, and by an image depth D of 5 where each depth represents the mean reflectance value aggregation for each pixel of 30 subsequent wavelength-bands of a total of 150 bands. Intending to reduce high computational complexity and memory consumption, which is commonly known for 3DCNN approaches, only one filter channel (C) has been applied.

Table 1. Architecture of the proposed SpectralNet3D.

L	Type	Feature Map	ksz	kd	Act	Drop
0	Input	1x5x284x284				
1	Conv3d	1x4x278x278	7	2	LReLU	N
2	Conv3d	1x3x272x272	7	2	LReLU	N
3	Conv3d	1x2x266x266	7	2	LReLU	N
4	AvgPool3d	1x1x260x260	7	2	LReLU	N
5	FC	2048			LReLU	N
6	FC	1024			LReLU	N
7	FC	512			LReLU	Y
8	FC	4			LReLU	N

In order to extract dependent features between subsequent wavelength ranges (depth aggregations), convolutions in the actual sense of cross-correlations are conducted with a kernel depth (kd) of 2, stride 1 and padding 0. The extraction of spatial information is realized by kernel size (ksz) 7 since smaller or larger sizes turned out to produce less meaningful feature maps for the fully connected layers. After three depth convolutions and a pooling layer (L1-L4) the feature map gets flattened for the final fully connected (FC) layers. These FC layers (L5-L7) extract non-linear dependencies within the feature map by a Leaky-ReLU activation function and map them to 4 continuous target values (soil parameters) in the output layer (L8). For this approach Leaky-ReLU (LReLU) is the preferred choice before ReLU since it has shown that its slightly non-zero activations for negative inputs are most beneficial for this regression task. Furthermore, a dropout rate of 0.4 has been applied before the final output layer in order to introduce additional regularization besides the usage of L_2 weight decay for the avoidance of overfitting.

3.2.2. Training

Since the baseline approach uses the trainset-split of 0.866 as described in section 4.1 the training of SpectralNet3D uses the same split as it turned out to perform best among different randomly shuffled training splits. Before training the network weights of the Conv3D- and Linear Layers are initialized uniformly according to [18] in order to improve loss convergence. The residual errors between soil parameter predictions \hat{y} and ground truth y are represented by batch averaged residuals $a = (y - \hat{y})$. These residual errors a are

evaluated by the Huber loss [19][20] function:

$$\mathcal{L}_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & , \quad |a| \leq \delta \\ \delta(|a| - \frac{1}{2}\delta) & , \quad otherwise \end{cases} \quad (1)$$

Dependent on the chosen parameter δ , Huber loss uses MSE loss for $|a|$ smaller than δ and δ -weighted $|a|$ otherwise, which makes it robust to outliers and smooths fluctuations of small residual ranges at once. Since validation loss driven model selection is only possible with ground truth data the choice for the best model had to be done differently for challenge submission model trainings. Thus an average training loss is calculated for the latter case, which allows selecting a model, which is less overfitted compared to models with minimal training loss. Instead, the selected model tends to be chosen close to the epoch with the steepest training loss curve decrease. To avoid long training times, early stopping has been realized by checking a loss convergence threshold stopping when it has not exceeded for a certain number of epochs. Moreover, as soon the loss starts to converge an additional learning rate scheduler reduces the originally defined learning rate in order to detect further loss minima.

4. EXPERIMENTS

The proposed approach SpectralNet3D* (SpNet3D) from section 3.2 has been benchmarked versus RNNPlus (cf. section 3.1) and the baseline approach evaluated in section 2. This chapter describes the used datasets, the experimental setup as well as the performance metrics and benchmark results.

4.1. Experimental Setup and Data

Data for training and testing has been provided by the challenge. The training data set contains 1732 training samples with 4-dimensional soil parameter vectors provided after laboratory analysis as ground truth. In addition to that 1154 samples are provided for testing. The distribution of soil parameters is shown in Figure 2. However, no ground truth has been provided for the official test set images. Therefore, for our evaluation we use the training/validation split from the official starter kit of the challenge, with the first 1500 images of the training set for training and the following 232 images for validation. Every sample is an image patch composed of 150 contiguous hyperspectral bands covering the range of 462-942 nm, with a spectral resolution of 3.2 nm. In Figure 1 an example patch is shown, in comparison to the full image (left) and the masked patch for a single spectral band (middle).

The given HSI image patches include a mask for the image, where the images are trimmed. Even before applying the proposed masks, the sample patches come in a large variety of patch sizes due to the variation of agricultural parcel sizes.

Experiments for SpectralNet3D had been conducted on a paperspace instance Free-P5000 with hyperparameters bs 8,

lr (1e-3, 1e-4, 1e-5), wd 1e-5, seed 0 and epoch patience thresholds of 6 for lr scheduling and 20 as global patience as well as loss change patience threshold of 1e-2 for training without validation set ground truth. For reproducibility the code of our implementations is available on github ⁴.

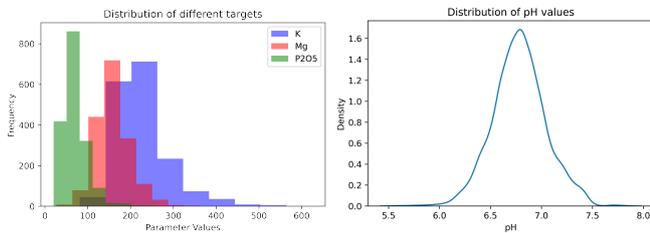


Fig. 2. Distribution of soil parameters Mg , K , and P_2O_5 (left) and pH values (right) in the training data.

4.2. Evaluation

For the quantitative evaluation of the submissions an individual score metric using MSE is given¹:

$$Score = \frac{\sum_{j=1}^4 \left(\frac{MSE_i}{MSE_i^{base}} \right)}{4}; \quad MSE_i = \frac{\sum_{j=1}^{|\psi|} (y_j - \hat{y}_j)^2}{|\psi|}$$

with test or validation set cardinality $|\psi|$. The MSE of the benchmarked approach is compared to the MSE of the baseline to create a score for evaluation. The baseline algorithm against which the performance is measured predicts the average value of each soil parameter obtained from the training set for all validation samples. The goal is to beat this baseline with a lowest possible score, which should be 0 in ideal case. For our choice of validation data see Section 4.1.

4.3. Results and Discussion

Considering the benchmark results in the following Table 2 one can observe that the proposed 3D convolutional approach is capable to improve the performance score of the baseline regressor on provided validation data. In addition to that, it could be shown that considering also the reflectance trend across different wavelengths of hyperspectral data could improve the soil parameter estimation performance. Comparing the best SpectralNet3D versions (with decreasing performance by v1 as best and v6 as worse version), Huber loss with $\delta = 0.15$ (v1) outperformed pure MAE loss(v3), SmoothL1 loss(v4) and MSE loss(v6) which is more vulnerable for outliers. Moreover, Adam(v1) has been preferred to AdamW(v2) for optimization. Different reflectance aggregation depths (v1: 5, v5: 8) demonstrate the varying importances of different wavelength resolutions for the prediction of each particular soil parameter (e.g. pH v1: 0.93 v5: 0.88).

⁴<https://github.com/ISDM-CAU-Kiel/ai4eo>

Table 2. Benchmark scores for soil parameters P_2O_5 , K , Mg and pH and their total average *Score*. (\downarrow : ideal score 0.0, best: bold, 2nd: italics).

Algorithm	Score \downarrow	P_2O_5 \downarrow	K \downarrow	Mg \downarrow	pH \downarrow
Baseline	1.0	1.0	1.0	1.0	1.0
RNNPlus	0.8081	1.001	0.4396	0.8143	0.9773
SpNet3D-v1	<i>0.8535</i>	0.9761	<i>0.6441</i>	<i>0.8540</i>	<i>0.9397</i>
SpNet3D-v2	0.8635	<i>0.9795</i>	0.6602	0.8637	0.9506
SpNet3D-v3	0.8688	1.0062	0.6759	0.8473	0.9457
SpNet3D-v4	0.8729	0.9826	0.6659	0.8772	0.9657
SpNet3D-v5	0.8951	0.9957	0.8017	0.8937	0.8893
SpNet3D-v6	0.9046	0.9809	0.7493	0.8656	1.0225

The 1D approach RNNPlus shows excellent results for potassium proving that potassium (K) can be inferred by considering only the mean reflectance, the results for magnesium are also superior. It shows a weakness in potassium and pH prediction for which it is not better than the baseline. However, for phosphorus pentoxide (P_2O_5) and pH the SpectralNet3D approach shows better results. Hence we argue, for these parameters the spatial and depth information of the images is important and using only the average reflectance would lose this information.

5. CONCLUSION

In this work we introduced a neural network regression approach for soil parameter estimation based 3D convolutional feature learning on hyperspectral image data, as well as a 1D RNN based approach using only average reflectance data. Experiments showed that both approaches can improve the prediction performance by utilizing the trend of reflectance across different wavelengths compared to the challenge baseline. However, performance on the validation split did not translate into better performance on the official challenge website on unreleased test data. We will investigate this issue in future work and consider the discussion on generalization capabilities of deep models in hyperspectral imaging in [21].

The 1D approach RNNPlus showed the best performance for potassium (K) and magnesium (Mg), while the 3D convolutional approach SpectralNet3D* showed the best performance for phosphorus pentoxide (P_2O_5) and pH values. Thus we conclude that both approaches have their strengths and weaknesses and for future work we propose a combination choosing the best method for each soil parameter. Approaches with a potentially better attention to the single wavelengths and different resolutions like e.g. SpectralFormer [22] might be capable to optimize the extracted feature maps. In addition to that, radial basis function (RBF) kernel models or variational latent distributions as part of the proposed neural architecture could be capable to even better cover the mixed soil parameter target distributions (see Figure 2).

6. REFERENCES

- [1] W Dean Hively, Gregory W McCarty, James B Reeves, Megan W Lang, Robert A Oesterling, and Stephen R Delwiche, "Use of airborne hyperspectral imagery to map soil properties in tilled agricultural fields," *Appl. Environ. Soil Sci.*, vol. 2011, 2011.
- [2] Naveen JP Anne, Amr H Abd-Elrahman, David B Lewis, and Nicole A Hewitt, "Modeling soil parameters using hyperspectral image reflectance in subtropical coastal wetlands," *Int. J. Appl. Earth Obs. Geoinf.*, vol. 33, pp. 47–56, 2014.
- [3] Chen Li, GW Zhang, ZG Zhou, WQ Zhao, YL Meng, BL Chen, and YH Wang, "Hyperspectral parameters and prediction model of soil moisture in coastal saline," *J. Appl. Ecol.*, vol. 27, no. 2, pp. 525–531, 2016.
- [4] Anita Bayer, Andreas Müller, Martin Bachmann, Sebastian Rößler, and Sebastian Weide, "Quantitative derivation of key soil parameters on the basis of hyperspectral remote sensing data," in *Proceedings of Hyperspectral Workshop 2010*. ESTEC, 2010.
- [5] Zouhaier Ben Rabah, Hedi Garbia, Emna Karray, Kais Tounsi, Abdelaziz Kallel, and Basel Solaiman, "Hyperspectral analysis for a robust assessment of soil properties using adapted pls method," *Advances in Remote Sensing*, vol. 8, no. 4, pp. 99–108, 2019.
- [6] Thomas Selige, Jürgen Böhner, and Urs Schmidhalter, "High resolution topsoil mapping using hyperspectral image and field data in multivariate regression modeling procedures," *Geoderma*, vol. 136, no. 1-2, pp. 235–244, 2006.
- [7] Zhen Li, Chenwei Deng, Baojun Zhao, Yibing Tian, and Yun Huang, "Hyperspectral inversion for soil moisture and temperature based on gaussian process regression," in *ICSIDP*, 2019, pp. 1–4.
- [8] Shengxiang Xu, Yongcun Zhao, Meiyang Wang, and Xuezheng Shi, "A comparison of machine learning algorithms for mapping soil iron parameters indicative of pedogenic processes by hyperspectral imaging of intact soil profiles," *Eur. J. Soil Sci.*, vol. 73, no. 1, pp. e13204, 2022.
- [9] Huan Yu, Bo Kong, Qing Wang, Xian Liu, and Xiangmeng Liu, "Hyperspectral remote sensing applications in soil: a review," *Hyperspectral Remote Sensing*, pp. 269–291, 2020.
- [10] Nicolas Audebert, Bertrand Le Saux, and Sébastien Lefèvre, "Deep learning for classification of hyperspectral data: A comparative review," *IEEE Geosci. Remote Sens. Mag.*, vol. 7, no. 2, pp. 159–173, 2019.
- [11] J Padarian, B Minasny, and AB McBratney, "Using deep learning to predict soil properties from regional spectral data," *Geoderma Regional*, vol. 16, pp. e00198, 2019.
- [12] Alberto Orgiazzi, Cristiano Ballabio, Panagiotis Panagos, Arwyn Jones, and Oihane Fernández-Ugalde, "Lucas soil, the largest expandable soil dataset for europe: a review," *Eur. J. Soil Sci.*, vol. 69, no. 1, pp. 140–153, 2018.
- [13] Shin Fujieda, Kohei Takayama, and Toshiya Hachisuka, "Wavelet convolutional neural networks," *arXiv preprint arXiv:1805.08620*, 2018.
- [14] Tanmay Chakraborty and Utkarsh Trehan, "Spectralnet: Exploring spatial-spectral waveletcnn for hyperspectral image classification," *arXiv preprint arXiv:2104.00341*, 2021.
- [15] Xin Wang, "Moving window-based double haar wavelet transform for image processing," *IEEE Transactions on Image Processing*, vol. 15, no. 9, pp. 2771–2779, 2006.
- [16] Mingyi He, Bo Li, and Huahui Chen, "Multi-scale 3d deep convolutional neural network for hyperspectral image classification," *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3904–3908, 2017.
- [17] Ignacio Oguiza, "tsai - a state-of-the-art deep learning library for time series and sequential data," Github, <https://github.com/timeseriesAI/tsai>, 2022.
- [18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, 2015, pp. 1026–1034.
- [19] Peter J. Huber, "Robust estimation of a location parameter," *Annals of Mathematical Statistics*, vol. 53, pp. 79–101, 1964.
- [20] "PyTorch Huber Loss implementation," <https://pytorch.org/docs/stable/generated/torch.nn.HuberLoss.html>.
- [21] Jakub Nalepa, Michal Myller, Marcin Cwiek, Lukasz Zak, Tomasz Lakota, Lukasz Tulczyjew, and Michal Kawulok, "Towards on-board hyperspectral satellite image segmentation: Understanding robustness of deep learning through simulating acquisition conditions," *Remote Sensing*, vol. 13, no. 8, 2021.
- [22] Danfeng Hong, Zhu Han, Jing Yao, Lianru Gao, Bing Zhang, Antonio Plaza, and Jocelyn Chanussot, "Spectralformer: Rethinking hyperspectral image classification with transformers," *IEEE Trans. Geosci. Remote Sens.*, 2021.

Chapter 9

Conclusion

With this thesis, we contributed to an outlier-driven machine learning research domain with a variety of methods. Therefore, this work introduced anomaly detection algorithms for point and group anomaly detection as well as approaches for evaluation and outlier robustness.

In chapter 3 we introduced with CoMadOut [33] an unsupervised point anomaly detection algorithm based on CoMAD PCA [41, 26] and adaptive outlier scoring methods. Thereby, we were able to demonstrate robustness towards outliers by distribution aligned principal components, which we utilized for a weighted scoring of outliers. For future works, we plan to estimate the ideal number of principal components and to investigate in meta-learners for model selection, as well as in the performance loss of achieving positive-semi-definiteness.

Furthermore, an unsupervised anytime point anomaly detection method, AnyCORE [32], is described in Chapter 4, providing domain experts with intermediate cluster-based outlier results.

In addition to that, we introduced in Chapter 5 GADFormer [34], a novel BERT [14]-based transparent transformer encoder architecture for un- and semi-supervised group anomaly detection on coordinate trajectories, which provides transparency regarding to an encoder block based separability of normal and abnormal trajectories to support the machine learning practitioner to find ideal hyperparameters. This approach could also be evaluated on image data, audio or text data. Moreover, transfer learning would bear potential for future work, e.g. by trying a GADFormer model pretrained on car trajectories also on bus trajectories.

In the next Chapter 6, we introduced a novel covariance vector representation [31] for distribution-based unsupervised group anomaly detection on bicycle rental movements, which allows to detect fee-based but frequently visited non-floating areas. A future work could be to also investigate in temporal group anomalies, revealing if visits of non-floating areas are only timely restricted. Furthermore, an application to other transportation services like eScooters might be of interest.

For testing anomaly detection algorithms in un- and semisupervised settings, we introduced in Chapter 7 an open anomaly benchmark framework (OAB) [35] to ease researchers of the anomaly detection domain their experiments on tabular and image data and to support reproducibility. Therefore, future works could focus on improving sampling by embedding clusters for

a better point difficulty balance in train and test set as well as to test the framework with a larger variety of algorithms and new datasets.

In Chapter 8 we introduced besides an RNNPlus approach a 3D convolutional architecture, SpectralNet3D [50], for supervised soil parameter prediction on hyperspectral image data based on outlier robust band range averages and Huber-Loss. A future work is to provide an ensemble of models with best prediction performance for each single soil parameter and to try latent distributions as part of the model architecture.

Bibliography

- [1] Charu C Aggarwal. *Outlier analysis second edition*. 2016.
- [2] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. “Ganomaly: Semi-supervised anomaly detection via adversarial training”. In: *Computer Vision—ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*. Springer. 2019, pp. 622–637.
- [3] Fabrizio Angiulli and Clara Pizzuti. “Fast outlier detection in high dimensional spaces”. In: *European conference on principles of data mining and knowledge discovery*. Springer. 2002, pp. 15–27.
- [4] Anna Beer, Jennifer Lauterbach, and Thomas Seidl. “MORE++: k-Means Based Outlier Removal on High-Dimensional Data”. In: *Similarity Search and Applications: 12th International Conference, SISAP 2019, Newark, NJ, USA, October 2–4, 2019, Proceedings 12*. Springer. 2019, pp. 188–202.
- [5] David M. Blei, A. Ng, and Michael I. Jordan. “Latent Dirichlet Allocation”. In: *J. Mach. Learn. Res.* 3 (2001), pp. 993–1022.
- [6] Azzedine Boukerche, Lining Zheng, and Omar Alfandi. “Outlier Detection: Methods, Models, and Classification”. In: *ACM Comput. Surv.* 53.3 (June 2020). ISSN: 0360-0300. DOI: 10.1145/3381028. URL: <https://doi.org/10.1145/3381028>.
- [7] Markus M. Breunig et al. “LOF: Identifying Density-Based Local Outliers”. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. SIGMOD ’00*. Dallas, Texas, USA: Association for Computing Machinery, 2000, pp. 93–104.
- [8] Guilherme O. Campos et al. “On the Evaluation of Unsupervised Outlier Detection: Measures, Datasets, and an Empirical Study”. In: *Data Min. Knowl. Discov.* 30 (July 2016), pp. 891–927.
- [9] Tanmay Chakraborty and Utkarsh Trehan. “SpectralNET: Exploring Spatial-Spectral WaveletCNN for Hyperspectral Image Classification”. In: *arXiv preprint arXiv:2104.00341* (2021).
- [10] Raghavendra et al. Chalapathy. “Group anomaly detection using deep generative models”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2018, pp. 173–189.
- [11] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly detection: A survey”. In: *ACM Comput. Surv.* 41.3 (July 2009). ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. URL: <https://doi.org/10.1145/1541880.1541882>.

- [12] Pierluca D’Oro et al. “Group Anomaly Detection via Graph Autoencoders”. In: 2019.
- [13] Hanbo Dai et al. “Detecting anomaly collections using extreme feature ranks”. In: *Data Mining and Knowledge Discovery* 29 (2015), pp. 689–731. URL: <https://api.semanticscholar.org/CorpusID:16764510>.
- [14] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, 2019, pp. 4171–4186.
- [15] Rémi Domingues et al. “A comparative evaluation of outlier detection algorithms: Experiments and analyses”. In: *Pattern Recognit.* 74 (Sept. 2018), pp. 406–421.
- [16] Andrew Emmott et al. “A Meta-Analysis of the Anomaly Detection Problem”. In: *arXiv: Artificial Intelligence* (2015).
- [17] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. “Knowledge Discovery and Data Mining: Towards a Unifying Framework”. In: *Knowledge Discovery and Data Mining*. 1996. URL: <https://api.semanticscholar.org/CorpusID:5708816>.
- [18] Hans Fischer. *A history of the central limit theorem: from classical to modern probability theory*. Vol. 4. Springer, 2011.
- [19] Ralph Foorthuis. “On the nature and types of anomalies: a review of deviations in data”. In: *International journal of data science and analytics* 12.4 (2021), pp. 297–331.
- [20] Shin Fujieda, Kohei Takayama, and Toshiya Hachisuka. “Wavelet convolutional neural networks”. In: *arXiv preprint arXiv:1805.08620* (2018).
- [21] Jorge Guevara, Stéphane Canu, and Roberto Hirata. “Support Measure Data Description for group anomaly detection”. In: *Knowledge Discovery and Data Mining*. 2015. URL: <https://api.semanticscholar.org/CorpusID:6771481>.
- [22] Riccardo Guidotti et al. “A survey of methods for explaining black box models”. In: *ACM computing surveys (CSUR)* 51.5 (2018), pp. 1–42.
- [23] D. M. Hawkins. *Identification of Outliers*. Monographs on Applied Probability and Statistics. Springer, 1980. ISBN: 978-94-015-3996-8. DOI: 10.1007/978-94-015-3994-4. URL: <https://doi.org/10.1007/978-94-015-3994-4>.
- [24] Xiang Jiang et al. “Improving point-based AIS trajectory classification with partition-wise gated recurrent units”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 4044–4051.
- [25] I.T. Jolliffe. *Principal Component Analysis*. New York: Springer, 1986.

- [26] Daniyal Kazempour, M. A. X. Hünemörder, and Thomas Seidl. “On coMADs and Principal Component Analysis”. In: *Similarity Search and Applications - 12th International Conference, SISAP 2019, Newark, NJ, USA, Proceedings*. 2019, pp. 273–280.
- [27] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).
- [28] Aditya Kuppa et al. “Finding Rats in Cats: Detecting Stealthy Attacks using Group Anomaly Detection”. In: *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)* (2019), pp. 442–449.
- [29] Tianyang Lin et al. “A survey of transformers”. In: *AI Open* 3 (2022), pp. 111–132. ISSN: 2666-6510. DOI: <https://doi.org/10.1016/j.aiopen.2022.10.001>. URL: <https://www.sciencedirect.com/science/article/pii/S2666651022000146>.
- [30] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. “Isolation-Based Anomaly Detection”. In: *ACM Transactions on Knowledge Discovery from Data* 6.1 (2012).
- [31] Andreas Lohrer, Johannes Josef Binder, and Peer Kröger. “Group Anomaly Detection for Spatio-Temporal Collective Behaviour Scenarios in Smart Cities”. In: *Proceedings of the 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science, IWCTS 2022, Seattle, Washington, 1 November 2022*. Ed. by Andy Berres, Kuldeep R. Kurte, and Haowen Xu. ACM, 2022, 14:1–14:4. DOI: 10.1145/3557991.3567801. URL: <https://doi.org/10.1145/3557991.3567801>.
- [32] Andreas Lohrer et al. “AnyCORE - An Anytime Algorithm for Cluster Outlier REmoval”. In: *Proceedings of the LWDA 2021 Workshops: FGWM, KDML, FGWI-BIA, and FGIR, Online, September 1-3, 2021*. Ed. by Thomas Seidl, Michael Fromm, and Sandra Obermeier. Vol. 2993. CEUR Workshop Proceedings. CEUR-WS.org, 2021, pp. 145–156. URL: <https://ceur-ws.org/Vol-2993/paper-14.pdf>.
- [33] Andreas Lohrer et al. “CoMadOut - A Robust Outlier Detection Algorithm based on CoMAD”. In: *Machine Learning* (May 2024). ISSN: 1573-0565. DOI: 10.1007/s10994-024-06521-2. URL: <https://doi.org/10.1007/s10994-024-06521-2>.
- [34] Andreas Lohrer et al. “GADformer: A Transparent Transformer Model for Group Anomaly Detection on Trajectories”. In: (2024). arXiv: 2303.09841 [cs.LG].
- [35] Andreas Lohrer et al. “OAB - An Open Anomaly Benchmark Framework for Unsupervised and Semisupervised Anomaly Detection on Image and Tabular Data Sets”. In: *2021 International Conference on Data Mining, ICDM 2021 - Workshops, Auckland, New Zealand, December 7-10, 2021*. IEEE, 2021, pp. 991–1000. DOI: 10.1109/ICDMW53433.2021.00129.

- [36] Edward McFowland, Skyler Speakman, and Daniel B. Neill. “Fast generalized subset scan for anomalous pattern detection”. In: *J. Mach. Learn. Res.* 14 (2013), pp. 1533–1561. URL: <https://api.semanticscholar.org/CorpusID:6831602>.
- [37] Krikamol Muandet and Bernhard Schölkopf. “One-Class Support Measure Machines for Group Anomaly Detection”. In: *ArXiv abs/1303.0309* (2013). URL: <https://api.semanticscholar.org/CorpusID:2007710>.
- [38] Martin Mühlfnzl. “Warum viele Gemeinden mit dem MVG-Mietrad-system unzufrieden sind”. In: *Süddeutsche Zeitung* (June 10, 2022). Online: 2024-02-19. URL: <https://www.sueddeutsche.de/muenchen/landkreismuenchen/landkreis-muenchen-mvg-rad-verkehr-unterschleissheim-ottobrunn-1.5601001>.
- [39] Joelle Pineau et al. “Improving Reproducibility in Machine Learning Research(A Report from the NeurIPS 2019 Reproducibility Program)”. In: *Journal of Machine Learning Research* 22.164 (2021), pp. 1–20. URL: <http://jmlr.org/papers/v22/20-303.html>.
- [40] Lukas Ruff et al. “Deep Semi-Supervised Anomaly Detection”. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL: <https://openreview.net/forum?id=HkgH0TEYwH>.
- [41] T. A. Sajesh and M. R. Srinivasan. “Outlier detection for high dimensional data using the Comedian approach”. In: *Journal of Statistical Computation and Simulation* 82.5 (2012), pp. 745–757.
- [42] Bernhard Schölkopf et al. “Estimating the support of a high-dimensional distribution”. In: *Neural computation* 13.7 (2001), pp. 1443–1471.
- [43] Mei-Ling Shyu et al. “A novel anomaly detection scheme based on principal component classifier”. In: *Proceedings of the IEEE foundations and new directions of data mining workshop*. IEEE Press, 2003, pp. 172–179.
- [44] Wanjuan Song, Wenyong Dong, and Lanlan Kang. “Group anomaly detection based on Bayesian framework with genetic algorithm”. In: *Information Sciences* 533 (2020), pp. 138–149. ISSN: 0020-0255. DOI: <https://doi.org/10.1016/j.ins.2020.03.110>. URL: <https://www.sciencedirect.com/science/article/pii/S0020025520302905>.
- [45] Edward Toth and Sanjay Chawla. “Group deviation detection methods: a survey”. In: *ACM Computing Surveys (CSUR)* 51.4 (2018), pp. 1–38.
- [46] Zhongqiu Wang et al. “Unsupervised learning trajectory anomaly detection algorithm based on deep representation”. In: *International Journal of Distributed Sensor Networks* 16.12 (2020), p. 1550147720971504.
- [47] Liang Xiong, Barnabás Póczos, and Jeff Schneider. “Group Anomaly Detection Using Flexible Genre Models”. In: *Proceedings of the 24th International Conference on Neural Information Processing Systems*. NIPS’11. Granada, Spain: Curran Associates Inc., 2011, pp. 1071–1079.

- [48] Liang Xiong et al. "Hierarchical Probabilistic Models for Group Anomaly Detection". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, Apr. 2011, pp. 789–797.
- [49] Rose Yu, Xinran He, and Yan Liu. "GLAD: Group Anomaly Detection in Social Media Analysis". In: *ACM Trans. Knowl. Discov. Data* 10.2 (Oct. 2015). ISSN: 1556-4681. DOI: 10.1145/2811268. URL: <https://doi.org/10.1145/2811268>.
- [50] Claudius Zelenka et al. "AI4EO Hyperview: A SpectralNet3D and RN-Plus Approach for Sustainable Soil Parameter Estimation on Hyperspectral Image Data". In: *2022 IEEE International Conference on Image Processing, ICIP 2022, Bordeaux, France, 16-19 October 2022*. IEEE, 2022, pp. 4263–4267. DOI: 10.1109/ICIP46576.2022.9897889. URL: <https://doi.org/10.1109/ICIP46576.2022.9897889>.
- [51] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. "A survey on unsupervised outlier detection in high-dimensional numerical data". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 5 (2012).