11th International Conference on Electrical, Electronics and Computer Engineering (IcETRAN), Nis, 3-6 June 2024

Comparative Analysis of Semi-Supervised Outlier Detection Algorithms

Matija Špeletić
University of Niš,
Faculty of Electronic Engineering
18000 Nis, Serbia
matija.speletic@elfak.ni.ac.rs

Stevica Cvetković
University of Niš,
Faculty of Electronic Engineering
18000 Nis, Serbia
stevica.cvetkovic@elfak.ni.ac.rs

Saša V. Nikolić
University of Niš,
Faculty of Electronic Engineering
18000 Nis, Serbia
sasa.nikolic@elfak.ni.ac.rs

Abstract—**Outlier detection plays a crucial role in identifying anomalous data points within datasets. While unsupervised methods have been widely used for this task, situations may arise where a dataset is known to contain no outliers for a specific task. In such cases, semi-supervised outlier detection algorithms become imperative. This paper presents a comprehensive analysis of multiple semi-supervised outlier detection techniques. The study evaluates these algorithms across diverse datasets, considering variations in feature dimensionality, sample size, and outlier prevalence. As a main contribution, this study provides valuable insights into the strengths and limitations of semi-supervised outlier detection algorithms, aiding researchers and practitioners in selecting suitable methods for outlier detection tasks.**

Keywords—outlier detection, anomaly detection, semi-supervised machine learning

I. Introduction

Outlier detection is a task of identifying a subset of data points that differs from other instances and is therefore assumed anomalous. An established approach for identifying anomalies within a dataset involves employing unsupervised methods such as local outlier factor (LOF) [1], angle-based outlier detection [2], and isolation forest [3]. However, situations may arise where we are confident that our dataset contains no outliers for a specific task. In such scenarios, it becomes prudent to explore alternative solutions tailored to such tasks, leading us to semi-supervised outlier detection algorithms. Semi-supervised outlier detection methods aim to identify anomalous data points in a dataset where only a portion of the data is labeled. We assume that our dataset solely comprises normal data samples (inliers). Therefore, our goal is to develop a model that can differentiate between new instances that follow the patterns of these normal samples and those that deviate as anomalies.

This study gives an exhaustive analysis of several semi-supervised outlier detection techniques across diverse datasets, explaining their efficacy and intricacies. Through a rigorous evaluation, the paper compares these methods, outlining their respective merits and limitations. Therefore, it provides valuable insights for selecting suitable methods in outlier detection tasks.

II. Related Work

A. *Dimensionality Reduction for Outlier Detection*

A methodology for outlier detection involves leveraging dimensionality reduction techniques. The core concept revolves around employing dimensionality reduction on a dataset comprising predominantly of inliers, thereby obtaining a condensed representation of the data. Subsequently, upon reconstructing the compressed data, outliers exhibit a notably higher reconstruction error compared to the inliers. While Principal Component Analysis (PCA) stands as a conventional choice for dimensionality reduction, autoencoders emerge as a more adaptable solution due to their ability to capture intricate data patterns. Autoencoders are proficient in learning the latent compressed representation of data, manifested through the output of the middle-hidden layer characterized by significantly fewer nodes than the input layer. The network is trained to minimize the reconstruction error, which in turn defines the outlier score. The use of neural networks for dimensionality reduction is discussed in [4], [5]. Studies have demonstrated that simplistic autoencoder architectures yield results akin to PCA [6]; however, deeper and more intricate architectures offer superior reconstruction capabilities. For instance, a deep autoencoder can distill a 784-pixel image into merely six real numbers, a feat unattainable with PCA alone [5]. This heightened capacity renders autoencoders exceptionally adept and precise for outlier detection tasks.

B. *Local Outlier Detection via Graph Neural Networks*

Local outlier methods like LOF [1] and DBSCAN [7] have gained popularity owing to their simplicity, interpretability, and competitive performance compared to deep learning-based approaches. Despite their effectiveness, these methods often lack adaptability and struggle with hyper-parameter optimization, limiting their performance on diverse datasets. To address these limitations, [8] introduces a novel method called LUNAR (Learnable Unified Neighborhood-based Anomaly Ranking). LUNAR aims to unify local outlier methods under a message passing framework, leveraging the capabilities of graph neural networks to enable learnability. Unlike traditional approaches, LUNAR constructs k-NN graphs for any tabular dataset, utilizing node distances as input and employing a learnable message aggregation function. Moreover, to train the model effectively, negative samples are introduced to prevent trivial solutions, thereby facilitating the learning of a decision boundary between normal and anomalous samples. By incorporating these advancements, LUNAR offers a promising approach to anomaly detection that addresses the shortcomings of existing methods, particularly in terms of adaptability and performance optimization.

C. *Isolation Forests for Semi-Supervised Outlier Detection*

Initially designed for unsupervised detection, isolation forests (iForest) [3] have demonstrated efficacy in semi-supervised outlier detection. Within an isolation forest, isolation trees, structured as binary decision trees, recursively delineate partitions within a dataset by randomly selecting attributes and partitioning the data accordingly until all instances are singularly isolated. This stochastic partitioning yields trees wherein paths to anomalies are notably shorter than those leading to normal data instances, attributed to the scarcity of anomalies and their distinctiveness in attribute values. Consequently, collective shorter paths to individual data samples across the forest signify a high likelihood of anomaly presence. Notably, iForest exhibits rapid convergence in detection performance with a minimal number of trees, alongside the ability to attain high detection efficiency with modest sub-sampling sizes. Moreover, iForest demonstrates scalability to address extensive datasets and high-dimensional problems characterized by numerous irrelevant attributes.

Furthermore, the original paper proposing isolation forests underscores their applicability in semi-supervised tasks. Empirical results indicate a marginal decrease in detection performance when anomalies are excluded from training data, as evidenced by small reductions in AUC (Area Under the Curve). However, employing larger sub-sampling sizes mitigates this reduction, yielding AUC recoveries when anomalies are omitted from training.

III. Evaluating Semi-Supervised Outlier Detection Methods

This section delves into investigating semi-supervised outlier detection algorithms across diverse datasets, enlightening their strengths and weaknesses across various use cases.

A. *Methodology*

When assessing different outlier detection algorithms like iForest, Autoencoder, and LUNAR, the first step is to divide the dataset into separate training and testing subsets. For semi-supervised outlier detection techniques, it's crucial to ensure that the training subset exclusively comprises inliers (i.e., normal data instances), while the testing subset incorporates both inliers and outliers. To ensure this delineation, a stratified 70-30 partitioning scheme is employed, predicated upon the classification labels present in the dataset. Subsequently, outlier instances are excised from the training subset, thereby yielding a testing subset characterized by a consistent ratio of outliers to normal instances as observed in the original dataset. This approach is essential for obtaining statistically meaningful outcomes.

For each method under examination, the initial phase includes training the model utilizing the designated training set, which exclusively comprises inliers. Subsequently, these trained models are employed to predict outlier scores. Through rigorous comparison with ground truth labels, we proceed to compute the Receiver Operating Characteristic Area Under the Curve (ROC AUC) score and the accuracy, thereby quantifying the effectiveness of the model. In order to conduct a comprehensive evaluation of model performance, we systematically measure both the training duration and the inference time, the latter pertaining to the processing time required for the test set evaluation.

In all experimental evaluations, the *pyOD* [10] outlier detection library in Python was employed. Specifically, the iForest algorithm leverages the *scikit-learn* [11], [12] implementation as its underlying framework, while the Autoencoder model utilizes *TensorFlow* framework [13] for its implementation.

B. *Model architectures*

The iForest configuration comprises 100 isolation trees with a subsampling size of 256. The Autoencoder architecture consists of seven hidden layers, with six layers housing 16 neurons each, and a central layer containing 4 neurons, all utilizing the ReLu activation function. Training the Autoencoder will be executed over 25 epochs. Additionally, the LUNAR algorithm will be employed with the parameter $k = 5$. Uniformity will be maintained across all datasets by employing identical model architectures and hyperparameters.

C. *Benchmark Datasets*

The previously mentioned outlier detection methods are assessed using diverse datasets sourced from the *ADBench* [9] benchmark repository. The subset of *ADBench* datasets used for this evaluation is shown in Table 1. These datasets contain labeled samples denoting both anomalous and normal instances. Notably, the datasets exhibit variations in feature dimensionality, sample size, and outlier prevalence. This experimental design facilitates a comprehensive evaluation of algorithmic performance across a spectrum of outlier detection scenarios.

TABLE I. Benchmark Datasets Used for Evaluation

Dataset name	*# samples*	*# features*	*% anomalies*
fault	1941	27	34.67
Ionosphere	351	32	35.89
landsat	6435	36	20.71
letter	1600	32	6.25
magic	19020	10	35.16
mammography	11183	6	2.32
mnist	7603	100	9.21
musk	3062	166	3.11
optdigits	5216	64	2.88
PageBlocks	5393	10	9.46
pendigits	6870	16	2.27
Pima	768	8	34.9
satimage-2	5803	36	1.22
SpamBase	4207	57	39.91
Cardiotocography	2114	21	22.04

IV. Results and Discussion

This section presents results for various metrics and delves into the mechanisms underlying the performance of different algorithms across diverse benchmark datasets.

A. *Quantitative Results*

Figure 1 depicts a schematic representation of Receiver Operating Characteristic Area Under the Curve (ROC AUC) scores obtained from an evaluation of various semi-supervised outlier detection algorithms across diverse benchmark datasets. Notably, the LUNAR algorithm demonstrates commendable and consistent performance across all datasets, emerging as the top performer in the majority of cases. While the Autoencoder algorithm exhibits competitive performance akin to LUNAR, occasionally surpassing it, particularly evidenced in the Cardiotocography dataset, it also demonstrates notable performance discrepancies, evident in datasets such as: *optdigits*, *letter*, *landsat*, and *fault*. Conversely, the iForest algorithm generally yields inferior results compared to LUNAR, albeit maintaining relative competitiveness. A noteworthy observation is the marked consistency of iForest's performance across datasets, contrasting with the fluctuating performance of the Autoencoder, where iForest achieves comparable results to LUNAR in instances where the Autoencoder falters.

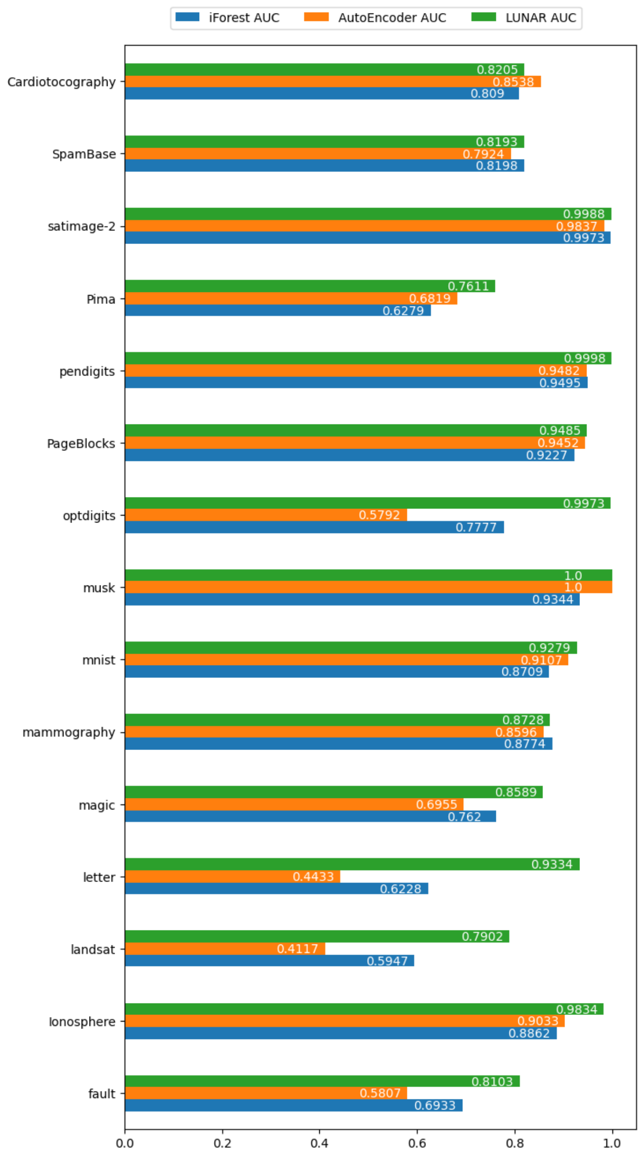

Fig. 1. ROC AUC scores of different outlier detection algorithms on multiple benchmark datasets.

Table 2 presents the F1 and Accuracy scores for the specified algorithms. It is apparent that all algorithms demonstrate comparable performance across various evaluation metrics. Notably, the F1 score highlights instances where the Autoencoder and iForest algorithms exhibit significant shortcomings, particularly evident in datasets such as *letter* and *optdigits*.

TABLE II. F1 AND ACCURACY SCORES FOR DIFFERENT OUTLIER DETECTION METHODS.

Dataset	**iForest**		**Autoencoder**		**LUNAR**	
	F1	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**
fault	0.551	0.636	0.462	0.609	0.613	0.736
Ionosphere	0.700	0.717	0.705	0.708	0.792	0.811
landsat	0.354	0.707	0.224	0.669	0.532	0.806
letter	0.081	0.858	0.033	0.879	0.417	0.942
magic	0.605	0.665	0.540	0.636	0.712	0.788
mammography	0.322	0.964	0.292	0.961	0.387	0.973
mnist	0.441	0.875	0.556	0.887	0.633	0.931
musk	0.278	0.943	0.644	0.965	0.795	0.984
optdigits	0.047	0.948	0.000	0	0.793	0.985
PageBlocks	0.527	0.877	0.614	0.901	0.708	0.944
pendigits	0.369	0.960	0.407	0.969	0.904	0.995
Pima	0.498	0.563	0.571	0.675	0.576	0.502
satimage-2	0.656	0.988	0.600	0.986	0.615	0.986
SpamBase	0.688	0.698	0.682	0.683	0.707	0.753
Cardiotocography	0.543	0.751	0.581	0.784	0.578	0.830

It is important to note that unlike the iForest and LUNAR algorithms, which typically demonstrate robust performance with minimal hyperparameter tuning, the Autoencoder method necessitates tailored construction for each specific task. In this study, the Autoencoder utilized consists of seven hidden layers, with six layers comprising 16 neurons each and the central layer containing four neurons. Particularly on datasets with a higher dimensionality, employing a larger Autoencoder with increased neuron count in the central layer may yield superior outcomes. This inherent sensitivity to neural network architecture poses a limitation of the Autoencoder method, as it may excel on certain datasets while underperforming on others, emphasizing the necessity for meticulous architecture selection. In contrast, Isolation Forests and LUNAR offer the advantage of consistent and satisfactory performance across diverse scenarios, obviating the need for extensive hyperparameter experimentation.

B. Time Performance Evaluation

The training and testing were performed exclusively using the CPU, without the aid of hardware accelerators. Table 3 gives the training durations of various algorithms across diverse datasets. Notably, the training time for the iForest algorithm is markedly trivial compared to that of both the Autoencoder and LUNAR models. This discrepancy stems from the isolation forest's linear time complexity, tailored to efficiently process expansive datasets while maintaining superior performance. Conversely, the autoencoder and LUNAR methods necessitate substantially longer training periods, surpassing 30 seconds for certain instances. Typically, Autoencoders demanded less time than LUNAR in these tests, although this is dependent the specific architecture employed.

Another crucial performance metrics to consider is the inference (i.e. test) time, denoting the duration required to process the test dataset, i.e., the time taken to discern whether a new instance qualifies as an outlier. These times are displayed in Table 3.

TABLE III. TRAIN AND TEST TIME FOR DIFFERENT OUTLIER DETECTION METHODS (THE TRAIN TIME IS GIVEN IN SECONDS AND THE TEST TIME IN MILLISECONDS).

Dataset	iForest		Autoencoder		LUNAR	
	Train (s)	**Test (ms)**	**Train (s)**	**Test (ms)**	**Train (s)**	**Test (ms)**
fault	0.09	4	4.53	73.9	3.80	20.5
Ionosphere	0.11	2	3.03	61.4	1.52	7.1
landsat	0.16	9	7.56	120.1	11.49	24.4
letter	0.12	3	4.01	72.3	4.14	4
magic	0.19	21	12.89	227.7	28.26	352.8
mammography	0.15	13	11.16	170.4	32.46	123.9
mnist	0.24	16	12.98	177.1	23.26	56
musk	0.18	7.5	8.70	129	10.63	19
optdigits	0.18	10	9.27	140.4	17.16	28.1
PageBlocks	0.18	13.2	9.58	175.6	16.64	41
pendigits	0.21	15.5	8.67	125.8	16.09	25
Pima	0.11	2	3.26	64.1	2.27	2
satimage-2	0.13	8	8.63	133.2	19.03	25.8
SpamBase	0.15	7.5	6.68	133.4	9.28	14
Cardiotocography	0.17	5	5.61	122.7	6.68	6

As anticipated, the iForest method consistently exhibits the shortest inference time (test time). Conversely, the Autoencoder tends to require the most time across various scenarios, attributable to the depth of the neural network utilized. Analogous to earlier observations, distinct architectures may yield diverse performance outcomes. Notably, the LUNAR model demonstrates remarkable speed across numerous datasets, with inference times often comparable to those of the isolation forest. However, on larger datasets like the *magic* dataset, the inference time for outlier scores notably escalates, with even the Autoencoder outperforming in terms of efficiency.

V. CONCLUSION

In this paper, we embarked on exploring semi-supervised outlier detection algorithms, recognizing the importance of tailored solutions for tasks where outliers are absent within the available dataset. We summarized various methods for outlier detection, that work on problems where the knowledge of normal data samples is available. Our investigation focused on several popular methods such as iForest, Autoencoders, and LUNAR.

Analyzing the results obtained, it became evident that the LUNAR method consistently performed exceptionally well across diverse datasets and application domains, showcasing its efficacy in outlier detection tasks. However, a notable limitation surfaced - its extensive training duration compared to other methods, which can pose a practical challenge in real-world applications. On the other hand, autoencoders demonstrated promise in delivering good results, although with a caveat: their sensitivity to model architecture demands meticulous customization for optimal performance tailored to specific datasets. Isolation forest method emerged as a reliable contender, providing consistently favorable results across various scenarios. Although it often fell short compared to LUNAR, its unparalleled speed and effectiveness, particularly with large datasets, underscore its significance in practical settings.

For future work, there is a promising avenue in exploring additional semi-supervised algorithms across a broader spectrum of datasets. Furthermore, delving deeper into refining model architectures, particularly focusing on autoencoders, holds potential for enhancing performance and adaptability in outlier detection tasks. By undertaking this line of research, we aim to further advance the field of semi-supervised outlier detection and address existing challenges to bolster its applicability in diverse domains.

ACKNOWLEDGEMENT

This work was supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia grant number 451-03-65/2024-03/200102.

REFERENCES

[1] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “LOF: identifying density-based local outliers,” in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[2] 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*, 2008, pp. 444–452.

[3] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in *2008 eighth ieee international conference on data mining*, IEEE, 2008, pp. 413–422.

[4] R. Hecht-Nielsen, “Replicator neural networks for universal optimal source coding,” *Science*, vol. 269, no. 5232, pp. 1860–1863, 1995.

[5] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[6] P. Baldi, “Autoencoders, unsupervised learning, and deep architectures,” in *Proceedings of ICML workshop on unsupervised and transfer learning*, JMLR Workshop and Conference Proceedings, 2012, pp. 37–49.

[7] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, and others, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *kdd*, 1996, pp. 226–231.

[8] A. Goodge, B. Hooi, S.-K. Ng, and W. S. Ng, “Lunar: Unifying local outlier detection methods via graph neural networks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 6737–6745.

[9] S. Han, X. Hu, H. Huang, M. Jiang, and Y. Zhao, “ADBench: Anomaly Detection Benchmark,” in *Neural Information Processing Systems (NeurIPS)*, 2022.

[10] Y. Zhao, Z. Nasrullah, and Z. Li, “PyOD: A Python Toolbox for Scalable Outlier Detection,” *J. Mach. Learn. Res.*, vol. 20, no. 96, pp. 1–7, 2019.

[11] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[12] L. Buitinck *et al.*, “API design for machine learning software: experiences from the scikit-learn project,” in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[13] Martín Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.” 2015. [Online]. Available: https://www.tensorflow.org/