Unsupervised Artificial Neural Networks for Outlier Detection in High-Dimensional Data

Daniel Popovic[0000−0002−5498−1516], Edouard Fouché[0000−0003−0157−7648], and Klemens Böhm[0000−0002−1706−1913]

Karlsruhe Institute of Technology (KIT), Germany
popovic@cognitana.com, {edouard.fouche, klemens.boehm}@kit.edu

This is a post-peer-review, pre-copyedit version of an article published in ADBIS 2019: Advances in Databases and Information Systems. The final authenticated version is available online at: https://doi.org/10.1007/978-3-030-28730-6_1

Abstract. Outlier detection is an important field in data mining. For high-dimensional data the task is particularly challenging because of the so-called "curse of dimensionality": The notion of neighborhood becomes meaningless, and points typically show their outlying behavior only in subspaces. As a result, traditional approaches are ineffective. Because of the lack of a ground truth in real-world data and of a priori knowledge about the characteristics of potential outliers, outlier detection should be considered an unsupervised learning problem. In this paper, we examine the usefulness of unsupervised artificial neural networks – autoencoders, self-organising maps and restricted Boltzmann machines – to detect outliers in high-dimensional data in a fully unsupervised way. Each of those approaches targets at learning an approximate representation of the data. We show that one can measure the "outlierness" of objects effectively, by measuring their deviation from the learned representation. Our experiments show that neural-based approaches outperform the current state of the art in terms of both runtime and accuracy.

Keywords: Unsupervised Learning · Outlier Detection · Neural Networks

1 Introduction

Outliers are objects that deviate significantly from others as to arouse the suspicion that a different mechanism has generated them [17]. The search for outliers has interested researchers and practitioners for many years, with applications such as the detection of fraud or intrusions, and medical diagnosis.

In real-world use cases, the characteristics of outliers are unknown beforehand. One can only obtain a ground truth with the help of domain experts, who produce explicit labels on the nature of data points. However, generating this ground truth is costly or even impossible. In high-dimensional spaces in particular, objects can be outlying in unexpected ways, which the expert does not notice during inspection. For example, in aircraft fault diagnostics, thousands of sensors collect huge amounts of in-flight data. The sensors not only collect airplane

data (accelerometer, speed sensor, voltage sensors, etc.) but also environmental or weather data (thermocouple, pressure sensors, etc.) [43]. Since the space of all valid sensor-value combinations is unknown a priori, it is impossible to discern between normal and abnormal instances. Thus, one cannot train a classifier in a supervised way, or even obtain a set of instances labelled as "normal". The absence of training data results in a fully unsupervised learning problem.

When the data is high-dimensional, i.e., has hundreds of dimensions, traditional outlier detectors do not work well at all. This is due to a number of effects summarized as the "curse of dimensionality" [3,4]. There exists a number of outlier detectors, which are robust against high dimensionality: ABOD and FastABOD [30] use the angle between data objects as a deviation measure. HiCS [24] and approaches by Aggarwal *et al.* [1], Kriegel *et al.* [29], Müller *et al.* [36] and Nguyen *et al.* [38,39] propose to assess the outlierness of data points only in low-dimensional projections of the full space. While these approaches do solve the problem in high-dimensional spaces to some extent, they often come with high computational complexity and unintuitive parameters.

Several studies propose to use artificial neural networks (ANNs) for outlier detection [23,37,18,15,10,34,9]. However, these studies have only considered relatively low-dimensional settings, i.e., fewer than 100 dimensions. Thus, the performance of these approaches in high-dimensional spaces is so far unknown. Next, they often treat outlier detection as a supervised or semi-supervised problem, which contradicts our view on it as unsupervised.

According to Bishop [5], discrepancies between the data used for training and testing is one of the main factors leading to inaccurate results with neural networks. ANN-based outlier detection approaches make use of this erroneous behavior on novel data, interpreting the deviations from the expected results – or "errors" – of the neural networks as an indication for "outlierness". The idea is to train the network to learn a good representation of the majority of the data objects. Since outliers are assumed to be "few and different"[33], the hypothesis is that they do not fit the representation learned by the model. Thus, one can detect them by measuring the respective error of the neural network.

To our knowledge, this study is the first to describe and compare the specifics of a range of ANN models for such an unsupervised detection of outliers in high-dimensional spaces, together with an extensive empirical evaluation. We articulate our contributions as follows:

- **We describe the principles of three unsupervised ANN-based approaches for outlier detection**, either based on autoencoders (AEs), self-organizing maps (SOMs) or restricted Boltzmann machines (RBMs).
- **We study the effects of different parameter settings for the ANN-based approaches empirically**. Based on this evaluation, we recommend parameter values to tune each approach for the outlier detection task.
- **We compare our approaches to state-of-the-art outlier detectors, using 26 real-world data sets**. The results show a significantly better detection quality of AE and SOM on most data sets for a reduced runtime.

– **We release our implementation** on GitHub[1], to ensure the reproducibility of the experiments and further use of the algorithms.

Paper outline: Section 2 features the related work. Section 3 presents our adaption of three families of ANNs for outlier detection. Section 4 is our evaluation. Section 5 summarizes our results and possible further research questions.

2 Related Work

2.1 High-Dimensional Outlier Detection

Numerous approaches exist to detect outliers in high-dimensional spaces. One can classify them as density-based [1,36], deviation-based [29,35], distance-based [38] or angle-based [30]. Other approaches, such as Isolation Forest [33], use decision tree ensembles to identify outliers. Alternatively, HiCS [24] decouples the search for subspaces from the actual outlier ranking.

Although methods explicitly targeting at high-dimensional data yield better results than most traditional methods, they come with certain drawbacks. First, most methods have high computational complexity and therefore do not scale well to large data sets. Second, each of these detectors requires at least one parameter, and the detection quality strongly depends on the parameter values for a given data set, as observed in [8]. There is little or no indication on how to choose suitable parameter values for these detectors. We in turn provide recommendations for suitable values for the approaches we investigate.

As an example, the time complexity of the original ABOD algorithm is in $O(n^3)$, which is not efficient with large data sets [30]. Even though there exists a faster version, FastABOD [30], with complexity in $O(n^2 + nk^2)$, the complexity remains quadratic with the number of objects, and determining a "good" value for parameter k is not straightforward.

2.2 ANNs for Outlier Detection

There also exists a number of approaches based on artificial neural networks. Japkowicz *et al.* [23] and Hawkins *et al.* [18] propose approaches based on the Autoencoder, sometimes named *Replicator Neural Network*. The work was followed by [12,34,9]. Muñoz and Muruzábal [37] were the first to propose an approach based on SOM and Sammon's mapping [45]. Fiore *et al.* [15] use the RBM to detect outliers in a semi-supervised way, as well as [10]. However, each of these contributions has at least one of the following issues:

– **Low-dimensional:** The evaluation of the approach is restricted to data with few dimensions, i.e., typically less than 100.
– **(Semi-)supervised:** Outlier detection operates only in a supervised or semi-supervised way, so that the unsupervised setting remains unaddressed.

[1] `https://github.com/Cognitana-Research/NNOHD`

– **Specialized:** The approach is tailored to a specific scenario, such as time series, or assumes the availability of prior knowledge, i.e., it is not applicable to the general outlier detection problem.

To our knowledge, the effectiveness of ANNs for general outlier detection is unknown so far in high-dimensional spaces under the unsupervised setting. In this work we correct this and compare ANN-based models to the state-of-the-art.

3 ANN-based Approaches

3.1 Requirements & General Idea

Any method mentioned in Section 2 comes with at least one disadvantage for high-dimensional outlier detection. Given this, we formulate requirements on new methods for the detection of outliers in high-dimensional spaces:

– **R1: Accuracy.** Superior outlier detection results in high-dimensional spaces, compared to existing approaches.
– **R2: Runtime.** Low computational burden, which allows the deployment of the method on high-dimensional data.
– **R3: Parameterization.** Small range of possible parameter values. Ideally, one should be able to derive default parameter values for high outlier-detection quality or recommendations for parameter-value selection a priori.

We show that ANN-based models fulfill these requirements. We only consider unsupervised ANN models and focus on the three main families: autoencoders (AEs), self-organizing maps (SOMs), and restricted Boltzmann machines (RBMs).

Unsupervised approaches have in common that they learn a representation of the data. One can measure the deviation of each object from this representation and use it as a score of the "outlierness" of this object. The implicit assumption here is that outliers are "few and different", so that they do not fit the learned representation and have a greater outlier score OS:

$$OS(x_{inlier}) < OS(x_{outlier}) \tag{1}$$

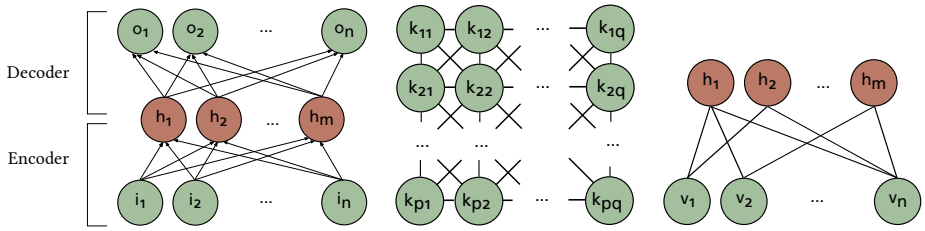
This score in turn determines a "confidence" that a given point x is an outlier: The higher the score, the higher this confidence. Note that, as a standard preprocessing step, one may scale the values of each dimension for each data set to $[0, 1]$. This limits the effect of different scales in different dimensions.

3.2 Autoencoder

Model Description The autoencoder (AE) is a multi-layer neural network that learns a lower-dimensional representation of a data set, from which this data can be reconstructed approximately [6,21]. To achieve this, the AE has an input and an output layer, with a number of neurons n that matches the number of dimensions in the data set, and one or more hidden layers with different

numbers of neurons m_i, $m_i \neq n$. It is a combination of an *encoder* part that transforms the data into a representation called the *code* and a *decoder* part that transforms the code back to the original data space. Figure 1(a) graphs the typical AE architecture.

In general, no transformation exists that leads to a perfect reconstruction of all data objects. As a result, the output of the AE is an approximate reconstruction of the input data.



(a) AE Architecture (b) SOM Architecture (c) RBM Architecture

Fig. 1. ANN Architectures

The encoder and decoder can be represented as functions $\mathbf{y} = e(\mathbf{x})$ and $\hat{\mathbf{x}} = d(\mathbf{y})$ with $\mathbf{x}, \hat{\mathbf{x}} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^m$, which are learned in order to minimize the sum of the squared error of the reconstruction

$$argmin_{e,d} \|\mathbf{x} - d\left(e(\mathbf{x})\right)\|^2 \tag{2}$$

Our Outlier Detection Approach For the outlier detection we use an AE with one hidden layer using the so-called ReLU activation function [16] for the hidden layer, and the sigmoid activation function for the output layer. Such a design choice is considered standard, as it alleviates the so-called vanishing gradient problem [22] and gives way to fast computation [31].

The number of neurons in the hidden layer is set to a fraction of the number of dimensions in the data set examined. We call this parameter the encoding factor ϵ. Learning the weights of the model is done via the widely-known back-propagation algorithm [25,32], with the AdaDelta gradient optimizer [51], in a number n^e of training epochs. Since the data is scaled to $[0, 1]$, we use the binary cross entropy loss function [44] between the input $\mathbf{i}$ and the output $\mathbf{o}$, defined as

$$l(\mathbf{i}, \mathbf{o}) = -(\mathbf{i} \ln \mathbf{o} + (1 - \mathbf{i}) \ln (1 - \mathbf{o})) \tag{3}$$

As stated above, the AE learns an approximated reconstruction of the input data. The expectation is that the reconstruction of "abnormal" objects will be less accurate than for "normal" objects. We use the outlier score OS_{AE} of an

object x as the Euclidean distance between its actual values x_j and its reconstruction $\hat{x}_j$, similarly as in [18]:

$$OS_{AE}(x) = \sum_{j=1}^{n} \sqrt{(x_j - \hat{x}_j)^2} \tag{4}$$

3.3 Self-Organising Maps

Model Description A Self-Organising Map (SOM), also known as Kohonen network, is an ANN traditionally used for dimensionality reduction and visualization of high-dimensional data [26,27]. It is a projection from an n-dimensional set of data objects to a low-dimensional, usually two- or three-dimensional, grid. A neuron with an n-dimensional weight vector is associated with each node in the grid. In the two-dimensional case, the SOM consists of a $p \times q$ neuron matrix with neurons described by their weights $w_{ij}, i \in \{1, \ldots, p\}, j \in \{1, \ldots, q\}$. Figure 1(b) illustrates the architecture of a SOM.

Training using a n-dimensional data set $\mathbf{X} = \{x_1, \ldots, x_m\}$ consists of n^e epochs with m training steps per epoch, in total $T = n^e * m$ training steps. We let $\mathbf{w_{ij}}(t)$ denote the weight vectors after the t^{th} training step, $\mathbf{w_{ij}}(0)$ the initial weight vectors, $\alpha(t)$ a learning rate decreasing with t and $h_{ij,cd}(t)$ a neighborhood function for neurons w_{ij} and w_{cd}, instantiated as a smoothing kernel whose width decreases with t. We use a Gaussian neighborhood kernel.

Training the SOM is done by finding for each data object x_k the neuron that has the closest distance, usually the Euclidean distance, to this data object, also called the *best matching unit* (BMU), and updating the weight vector of each neuron as follows:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(t) h_{ij,cd}(t)(x_k - w_{ij}(t)) \tag{5}$$

So the weight vectors of the BMU and its neighboring neurons in the grid are moved closer to the data object, and we repeat this process iteratively.

Our Outlier Detection Approach Our approach is based on the idea that the trained SOM forms a map that is adjusted to the majority of data objects. Outliers are assumed to be located farther away from their BMUs than inliers.

As is common in the literature, we use a 2-dimensional SOM with n rows and columns, where n is called the *topology size*. We initialize the weights by selecting the first two subspaces spanned by the first two eigenvectors of the correlation matrix, as in [11,2].

The number of training epochs n^e is a parameter that has an effect on outlier detection quality. The outlier score for the SOM, OS_{SOM}, is the Euclidean distance of a data object to its BMU. We refer to the BMU for object $\mathbf{x}$ as $bmu_{\mathbf{x}}$.

$$OS_{SOM}(x) = \sum_{j=1}^{n} \sqrt{(x_j - bmu_{x_j})^2} \tag{6}$$

3.4 Restricted Boltzmann Machine

Model Description The restricted Boltzmann machine (RBM) is a stochastic ANN that learns a probability distribution over a training data set. It is a special case of the Markov random field [49,20]. The RBM consists of a layer of n "visible" neurons v_i and a layer of m "hidden" neurons h_j that form a bipartite graph with a connection weight matrix $\mathbf{W}_{n\times m} = (w_{ij})_{n\times m}$, a bias vector $\mathbf{a} =< a_i >_{i\in\mathbb{N}^{[1,n]}}$ for the visible neurons and a bias vector $\mathbf{b} =< b_j >_{j\in\mathbb{N}^{[1,m]}}$ for the hidden neurons. The probability distribution is defined using the energy function

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{a}^T\mathbf{v} - \mathbf{b}^T\mathbf{h} - \mathbf{v}^T\mathbf{W}\mathbf{h} \tag{7}$$

which assigns a scalar energy to each configuration, i.e., to each pair of visible and hidden neuron values. A high energy for a configuration corresponds to a low probability of that configuration to appear in the model. The objective of the training is to find a configuration of weights and biases that lead to a high probability for the training data objects and a low probability for other data. So the energy for data objects from the training data set is sought to be minimized in the training process. Using the gradient descent algorithm to minimize this objective function would involve the computation of the expectation over all possible configurations of the input data object, which is not feasible in practice. Hence, the training usually is performed with the contrastive divergence algorithm [19] using Gibbs sampling, which approximates the gradient descent. This algorithm simplifies and speeds up training compared to gradient descent. It performs three learning steps on each training data object. First, all hidden units are updated in parallel from the training data object at the visible neurons. Then, the visible neurons are updated in parallel to get a reconstruction of the training data object. Finally, the hidden neurons are updated in parallel again. Figure 1(c) illustrates the architecture of a RBM.

Our Outlier Detection Approach We use a RBM with Gaussian visible neurons. As it assumes that the data is normally distributed, we standardize each dimension by subtracting the mean and dividing by the standard deviation as a preprocessing step. After the training, the energy is expected to be low for normal data objects and high for rare or unknown data objects. The outlier score for a data object, OS_{RBM}, is its so-called free energy:

$$OS_{RBM}(x) = -\sum_i a_i x_i - \sum_i \frac{x_i^2}{2} + \sum_i \ln \sum_{h_i} e^{b_i + \mathbf{w_i} x} \tag{8}$$

The proportion δ of hidden units w.r.t. visible units, and the number of training epochs n^e are free parameters. The share of the data set used for training is referred to as γ.

4 Evaluation

In this section, we pursue two separate evaluations. First, we evaluate the parameter ranges of each of our approaches on high-dimensional data. This leads to

the recommendation of "good" parameters. Second, we compare the approaches against the state of the art and evaluate them using high-dimensional data.

We implement the models in Python 3 using Keras, Tensorflow and the SOM implementation Somoclu [50]. All experiments run on a quad-core processor at 3.20 GHz with 8 GB RAM. As mentioned earlier, we publish the source code for our experiments on GitHub[2], to ensure reproducibility.

4.1 Parameter Selection

Campos *et al.* [8] compare various parametrized outlier detection approaches, with a large range of parameter values. Their results indicate that the entire parameter-value range is needed to achieve the highest outlier detection quality over different data sets, and that there exists no obvious way to find good parameter values a priori for a given data set.

To verify whether this applies to NN-based approaches as well, we investigate the range of parameter combinations for each approach, based on 26 data sets. Table 1 lists the characteristics of each data set in the corpus. It contains the same data sets as in [9] except for KddCup99, for which there are no exact results in [9], plus an assortment of high-dimensional data sets: Arrhythmia [8], InternetAds [8], ISOLET [14], MNIST and Musk [42]. Arrhythmia, InternetAds and ISOLET have several variants with different proportions of outliers. In this work we present the results for the variants with approximately 2% outliers. Because of the restricted number of pages, we present the evaluation results for the other variants in our GitHub repository, evaluating in total 26 data sets.

Table 1. List of evaluated data sets.

Data Set	Dimensions	Data Objects	Outliers	Outlier Ratio
Arrhythmia-2	259	248	4	1.61%
Cardio	21	1,831	176	9.61%
Ecoli	7	336	9	2.68%
InternetAds-2	1,555	1,630	32	1.96%
ISOLET-2	617	2,449	50	2.00%
Lympho	18	148	6	4.05%
MNIST	100	7,603	700	9.21%
Musk	166	3,062	97	3.12%
Optdigits	64	5,216	150	2.88%
P53	5,408	16,592	143	0.86%
Pendigits	16	6,870	156	2.27%
Seismic	11	2,584	170	6.58%
Thyroid	6	3,772	93	2.4%
Waveform	21	3,509	166	4.73%
Yeast	8	1,364	65	4.77%

[2] `https://github.com/Cognitana-Research/NNOHD`

We evaluate the goodness of each parameter combination on the whole dataset assortment. The aim of the evaluation is to derive parameters that lead to results which are *best on average*. To this end, we define a notion of deviation $\mathcal{D}_p$ that should ideally be minimized:

$$\mathcal{D}_p(\mathcal{A}) = \frac{\sum_{d \in D} (S_d^{\max} - S_d^p)}{|D|} \tag{9}$$

Intuitively, the deviation $\mathcal{D}_p$ of an algorithm $\mathcal{A}$ is the average difference between the best achievable score $S_d^{\max}$ over the parameters $p \in \mathcal{P}$ and the actual score S_d^p obtained with parameter combination p for all data sets $d \in D$. In the end, choosing the parameter combination minimizing $\mathcal{D}_p$ means maximizing the normalized average score for each data set in the assortment. Our hope is that those parameters will lead to good outlier detection on data sets that are not part of this assortment as well, so they can be useful to others.

We instantiate the score S as the commonly used ROC AUC. For the AE, we investigate encoding factors from 0.5 to 0.9 in steps of 0.1. For the SOM, we use quadratic maps with columns and rows from the range $\{1, \ldots, 20\}$. We test the RBM with values for δ, the number of hidden neurons as share of the visible neurons, of 0.1 to 0.9 in steps of 0.1, and for γ, the share of data used for training, of 0.1 to 0.9 in steps of 0.1. For all three approaches we use 10, 20, 50, 100 and 1,000 training epochs. The final result for a parameter combination is computed as the average of 20 runs. As there is no obvious way to select these parameter values a priori, we test all $p \in \mathcal{P}$ for each of our approaches in a brute-force fashion. This results in a total number of 10,500 experiments for the AE, 42,000 for the SOM, and 170,000 for the RBM.

Table 2 is an excerpt of the parameter evaluation of the AE, SOM and RBM approaches. The best parameter value for each approach is in boldface, the parameter values that are within 0.01 of the best result are in gray boldface. We publish the complete list in our GitHub repository. The evaluation leads to the following recommendations:

- For the Autoencoder:
  - Encoding factor $\epsilon = 0.8$
  - Number of training epochs $n^e = 20$
- For the Self-organizing map:
  - Topology size $n = 2$
  - Number of training epochs $n^e = 10$
- For the Restricted Boltzmann machine:
  - Proportion of hidden neurons $\delta = 0.8$
  - Number of training epochs $n^e = 100$
  - Proportion of training data $\gamma = 0.9$

Interestingly, the number of training epochs n^e minimizing D_p for the ANN-based approaches is relatively low, between 10 and 20 epochs for AE and SOM and 100 for RBM, and this observation is consistent even if we vary the encoding factor ϵ. This means that, in contrast to other application domains of neural

Table 2. Parameter evaluation of AE and SOM (excerpt).

Parameters			AE	SOM	Parameters			RBM
ϵ	n	n^e	$\mathcal{D}_{\epsilon,n^e}$	$\mathcal{D}_{n,n^e}$	δ	n^e	γ	$\mathcal{D}_{\delta,n^e,\gamma}$
0.6	2	10	0.0478	**0.0213**	0.7	100	0.5	0.1148
0.6	2	20	0.0411	0.0320	0.7	100	0.9	0.0750
0.6	2	100	0.0476	0.0214	0.7	1000	0.5	0.0940
0.6	2	1,000	0.0844	0.0214	0.7	1000	0.9	0.0724
0.7	3	10	0.0501	0.0306	0.8	100	0.5	0.1173
0.7	3	20	0.0412	0.0457	0.8	100	0.9	**0.0597**
0.7	3	100	0.0527	0.0327	0.8	1000	0.5	0.0905
0.7	3	1,000	0.0788	0.0323	0.8	1000	0.9	0.0701
0.8	4	10	0.0409	0.0390	0.9	100	0.5	0.1027
0.8	4	20	**0.0403**	0.0500	0.9	100	0.9	0.0734
0.8	4	100	0.0512	0.0535	0.9	1000	0.5	0.0868
0.8	4	1,000	0.0827	0.0545	0.9	1000	0.9	0.0769

networks such as image recognition, good outlier detection is feasible with low computational effort, as few training epochs are required.

We can also see that SOM achieves the best results for very low-dimensional maps, i.e., only 2 or 3 columns/rows, which also stands for a low computational burden. For the RBM we notice that the best results are achieved for a larger number of epochs, namely 100 and 1,000, while the robustness is rather stable over the number of hidden neurons. Finally, we see that the average deviation for the large majority of the parameter values is not greater than 5%. Thus, the performance of neural-based methods seems to be relatively independent from the chosen parameters, i.e., they fulfill Requirement **R3**.

4.2 Outlier Detection Quality Evaluation

We now compare our approaches to the state of the art. We consider RandNet [9], which is – to our knowledge – the most recently published ANN-based contribution. The authors did not publish their implementation nor enough information for reproducibility, so we simply compare to the same data sets except for the 41-dimensional KddCup99, and the baseline methods LOF [7], Hawkins [18], HiCS [24] and LODES [46] as in [9]. The authors also have set $k = 5$ for LOF and HiCS, which yields suboptimal results for these approaches. Investigating all parameter values k in the range $k \in 1, \ldots, 100$ for LOF and HiCS, we find that $k = 100$ leads to the best results on average. Thus, we repeat the evaluation on these data sets with $k = 100$ for a fair comparison. We further set HiCS parameters to $M = 50$, $\alpha = 0.1$ and *candidate_cutoff* $= 100$, which is in line with the recommendations by the authors [24]. For any competing algorithm, we use the implementation from ELKI [48].

Table 3 lists the ROC AUC values for the data sets in [9]. The best values in the table are highlighted in boldface, values within 1% of the best value are

highlighted in italics. We see that AE and SOM are at least competitive using the recommended parameter values. SOM even outperforms all approaches in three data sets. The RBM stands behind for most data sets, only having competitive results for the Pendigits and Thyroid data sets. Surprisingly, in a few cases (e.g., [Optdigits, LODES]), the score falls way below 0.5, i.e., it is worse than random guessing. We notice that this never occurs with neural-based approaches.

Table 3. ROC AUC Comparison for data sets used in [9] (in %).

Data Set	AE	SOM	RBM	RandNet	LODES	HiCS	LOF	Hawkins
Cardio	*92.10*	**93.01**	53.75	*92.87*	78.90	85.59	91.41	*92.36*
Ecoli	87.19	86.65	76.62	85.42	**91.81**	88.25	90.35	82.87
Lympho	90.33	*99.77*	58.57	*99.06*	78.16	92.94	**99.88**	98.70
Optdigits	71.00	71.93	48.90	*87.11*	2.66	37.94	38.94	**87.63**
Pendigits	66.53	**95.83**	91.25	93.44	87.88	72.78	51.51	89.81
Seismic	*71.88*	**71.99**	33.03	*71.28*	66.71	68.19	65.58	68.25
Thyroid	89.46	92.99	94.35	90.42	72.94	91.74	**96.31**	87.47
Waveform	59.22	69.39	60.35	70.05	62.88	71.82	**76.56**	61.57
Yeast	**83.81**	81.81	49.37	*82.95*	77.70	78.21	78.19	82.12

In Table 4, we observe similar results for the high-dimensional data sets Arrhythmia, ISOLET, MNIST, Musk and P53. We compare our approaches against LOF [7], HiCS [24], FastABOD [30], LoOP [28]. In addition, we use one-class SVM [47] and KNN Outlier [41] with $k = 1$ as a baseline. AE and SOM yield the highest ROC AUC for 4 of 6 data sets. For the ISOLET data set where LOF has the highest ROC AUC, AE and SOM are within reach of the best results. Only for InternetAds, which consists only of binary attributes, the ANN-based approaches fall behind the best results. Figure 2 graphs the ROC AUC comparisons for the high-dimensional data sets.

Table 4. ROC AUC comparison for high dimensional data sets (in %).

Data Set	AE	SOM	RBM	HiCS	LOF	FastABOD	LoOP	OC-SVM	KNN
Arrhythmia-2	**80.22**	76.33	49.04	50.56	76.74	76.84	75.00	77.66	71.88
InternetAds-2	43.36	66.12	46.93	**99.84**	71.64	76.49	77.14	64.18	81.23
ISOLET-2	96.85	*99.28*	92.28	79.71	**99.58**	93.09	98.23	92.05	94.66
MNIST	**82.06**	*81.07*	49.87	51.74	80.34	54.35	71.66	76.46	72.74
Musk	**100**	**100**	95.60	*99.60*	84.00	5.11	51.86	67.60	7.11
P53	60.63	**67.17**	64.76	62.09	61.99	62.92	61.99	61.27	62.56

Provost and Foster [40] argue that ROC AUC is not an appropriate performance measure for the classification of highly skewed data sets, which certainly is the case with outliers. Thus, we use the area under the precision-recall curve (PR

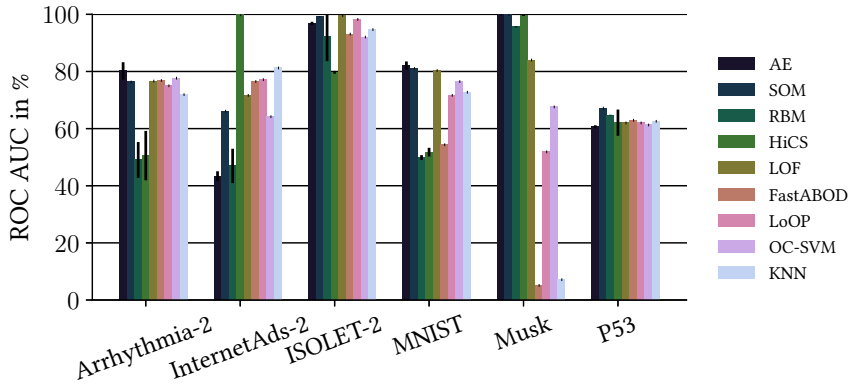


Fig. 2. ROC AUC for High-Dimensional Data Sets

AUC) as a complementary measure for the high-dimensional data-set evaluation [13]. Table 5 lists the corresponding PR AUC values.

Table 5. PR AUC comparison for high dimensional data sets (in %).

Data Set	AE	SOM	RBM	HiCS	LOF	FastABOD	LoOP	OC-SVM	KNN
Arrhythmia-2	**29.37**	27.94	1.66	1.64	3.83	3.98	3.51	4.90	3.04
InternetAds-2	1.58	32.78	1.91	**94.52**	34.86	32.79	37.09	24.83	32.24
ISOLET-2	44.45	66.93	29.51	4.47	**70.95**	29.54	51.78	25.65	42.91
MNIST	30.13	27.40	9.26	9.99	**33.95**	14.05	24.74	25.49	27.96
Musk	**100**	**100**	85.58	97.46	14.68	1.65	3.71	4.54	1.91
P53	1.04	*1.29*	1.20	1.25	1.06	1.13	1.06	1.06	**1.30**

As we can see, AE and SOM have the best results in terms of PR AUC for 2 out of 6 data sets and are close to the best results for the other data sets except for InternetAds. This also indicates that AE and SOM have fewer false positives in most data sets. A significant insight is that SOM, and to a certain degree also AE, deliver competitive results over all data sets, while all reference algorithms fall behind the top group by much, at least for some data sets. This stability of results is a great advantage for the SOM and the AE. The RBM again has a mixed performance: It is competitive on the ISOLET and Musk data sets, but close to guessing for the Arrhythmia and MNIST data sets. It is noticeable that the RBM yields results similar to random guesses for all sparse data sets that are evaluated, namely the Arrhythmia, InternetAds, Lympho, MNIST and Optdigits data sets.

4.3 Runtime Evaluation

We measure the execution time of our approaches. Each algorithm runs with the recommended parameter values.

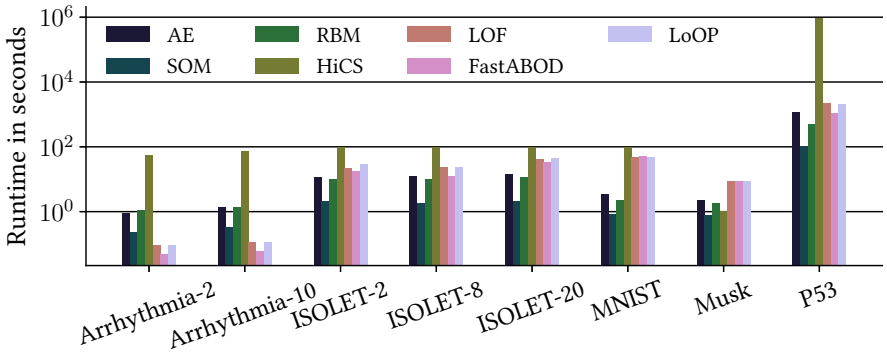


Fig. 3. Runtime comparison

Figure 3 graphs the average execution time for selected data sets with a logarithmic scale. We see that SOM is very fast for all data sets. This is particularly obvious for the P53 data set, which consists of 5,408 dimensions and 16,592 data objects. While the fastest of the compared algorithms needs more than 18 minutes, the SOM needs less than 2 minutes. The RBM with less than 9 minutes and the AE with less than 20 minutes still are very fast. Note that the runtimes were measured without GPU support. For the AE in particular, the runtime using a GPU would be much smaller.

We could not compare the runtime of our approaches with RandNet, because of the missing implementation. However, since it consists of an ensemble of up to 200 AEs with 3 hidden layers, it should be clear that it requires much more computational effort than any of our neural-based approaches.

5 Conclusions

This paper studies the application of ANN-based models, namely autoencoder (AE), self-organizing map (SOM) and restricted Boltzmann machine (RBM), to high-dimensional outlier detection. For each of these approaches, we propose to use a model-specific outlier score. Nonetheless, the scores have in common that they quantify in a fully unsupervised way the deviation from the expected output for each data point w.r.t. the learned model.

We evaluate the models on an assortment of high-dimensional data sets and compare the results to state-of-the-art outlier detection algorithms. The SOM

and AE approaches show superior performance in terms of detection quality (Requirement **R1**) and runtime (Requirement **R2**) compared to the state of the art. SOM clearly outperforms them all and yields very high result quality in large high-dimensional data sets. At the same time, the range of relevant parameter values (Requirement **R3**) for AE and SOM is significantly smaller than for the state-of-the-art algorithms.

All in all, this study also shows that "simple" is often better in the case of outlier detection. When used properly, well-known ANN-based approaches such as AE and SOM outperform recently proposed approaches for unsupervised outlier-detection tasks in high-dimensional data, both in terms of accuracy and runtime, while being less sensitive to parameter tuning.

In the future, it will be interesting to investigate whether the extension of the RBM to deep belief networks (DBNs) [20] leads to better results for this class of algorithms, since RBM has shown a relatively low detection quality. In this study, we have determined good parameter values for each approach in the general case, but finding the optimal values for each data set for the AE and SOM might improve performance even more. Thus, our goal would be to come up with a method to set parameters automatically in a data-driven way.

References

1. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. In: SIGMOD Conference. pp. 37–46. ACM (2001). https://doi.org/10.1145/376284.375668
2. Attik, M., Bougrain, L., Alexandre, F.: Self-organizing map initialization. In: ICANN (1). Lecture Notes in Computer Science, vol. 3696, pp. 357–362. Springer (2005). https://doi.org/10.1007/11550822_56
3. Bellman, R.E.: Dynamic Programming. Princeton University Press (1957)
4. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is "nearest neighbor" meaningful? In: ICDT. Lecture Notes in Computer Science, vol. 1540, pp. 217–235. Springer (1999). https://doi.org/10.1007/3-540-49257-7_15
5. Bishop, C.M.: Novelty detection and neural network validation. In: ICANN '93. pp. 789–794 (1993). https://doi.org/10.1007/978-1-4471-2063-6_225
6. Bourland, H., Kamp, Y.: Auto-association by multilayer perceptrons and singular value decomposition. Biological Cybernetics **59**(4), 291–294 (1988). https://doi.org/10.1007/BF00332918
7. Breunig, M.M., Kriegel, H., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: SIGMOD Conference. pp. 93–104. ACM (2000). https://doi.org/10.1145/335191.335388
8. Campos, G.O., Zimek, A., Sander, J., Campello, R.J.G.B., Micenková, B., Schubert, E., Assent, I., Houle, M.E.: On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min. Knowl. Discov. **30**(4), 891–927 (2016). https://doi.org/10.1007/s10618-015-0444-8
9. Chen, J., Sathe, S., Aggarwal, C.C., Turaga, D.S.: Outlier detection with autoencoder ensembles. In: SDM. pp. 90–98. SIAM (2017). https://doi.org/10.1137/1.9781611974973.11
10. Chen, Y., Lu, L., Li, X.: Application of continuous restricted boltzmann machine to identify multivariate geochemical anomaly. Journal of Geochemical Exploration **140**, 56–63 (2014). https://doi.org/10.1016/j.gexplo.2014.02.013

11. Ciampi, A., Lechevallier, Y.: Clustering large, multi-level data sets: An apporach based on kohonen self organizing maps. In: PKDD. Lecture Notes in Computer Science, vol. 1910, pp. 353–358. Springer (2000). https://doi.org/10.1007/3-540-45372-5_36
12. Dau, A.H., Ciesielski, V., Song, A.: Anomaly detection using replicator neural networks trained on examples of one class. In: SEAL. Lecture Notes in Computer Science, vol. 8886, pp. 311–322. Springer (2014). https://doi.org/10.1007/978-3-319-13563-2_27
13. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: ICML. ACM International Conference Proceeding Series, vol. 148, pp. 233–240. ACM (2006). https://doi.org/10.1145/1143844.1143874
14. Dua, D., Graff, C.: UCI machine learning repository (2019), `http://archive.ics.uci.edu/ml`
15. Fiore, U., Palmieri, F., Castiglione, A., Santis, A.D.: Network anomaly detection with the restricted boltzmann machine. Neurocomputing **122**, 13–23 (2013). https://doi.org/10.1016/j.neucom.2012.11.050
16. Hahnloser, R.R., Sarpeshkar, R., Mahowald, M.A., Douglas, R.J., Seung, S.H.: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. Nature **405**(6789), 947–951 (2000). https://doi.org/10.1038/35016072
17. Hawkins, D.M.: Identification of Outliers, Monographs on Applied Probability and Statistics, vol. 11. Springer Netherlands, Dordrecht (1980). https://doi.org/10.1007/978-94-015-3994-4
18. Hawkins, S., He, H., Williams, G.J., Baxter, R.A.: Outlier detection using replicator neural networks. In: DaWaK. Lecture Notes in Computer Science, vol. 2454, pp. 170–180. Springer (2002). https://doi.org/10.1007/3-540-46145-0_17
19. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation **14**(8), 1771–1800 (2002). https://doi.org/10.1162/089976602760128018
20. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation **18**(7), 1527–1554 (2006). https://doi.org/10.1162/neco.2006.18.7.1527
21. Hinton, G.E., Zemel, R.S.: Autoencoders, minimum description length and helmholtz free energy. In: NIPS. pp. 3–10. Morgan Kaufmann (1993), `http://papers.nips.cc/paper/798-autoencoders-minimum-description-length-and-helmholtz-free-energy`
22. Hochreiter, S.: The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems **6**(2), 107–116 (1998). https://doi.org/10.1142/S0218488598000094
23. Japkowicz, N., Myers, C., Gluck, M.A.: A novelty detection approach to classification. In: IJCAI. pp. 518–523. Morgan Kaufmann (1995), `http://ijcai.org/Proceedings/95-1/Papers/068.pdf`
24. Keller, F., Müller, E., Böhm, K.: Hics: High contrast subspaces for density-based outlier ranking. In: ICDE. pp. 1037–1048. IEEE Computer Society (2012). https://doi.org/10.1109/icde.2012.88
25. Kelley, H.J.: Gradient theory of optimal flight paths. ARS Journal **30**(10), 947–954 (1960). https://doi.org/10.2514/8.5282
26. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics **43**(1), 59–69 (1982). https://doi.org/10.1007/bf00337288

27. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, Springer Berlin Heidelberg, Berlin, Germany (1995). https://doi.org/10.1007/978-3-642-97610-0
28. Kriegel, H., Kröger, P., Schubert, E., Zimek, A.: Loop: local outlier probabilities. In: CIKM. pp. 1649–1652. ACM (2009). https://doi.org/10.1145/1645953.1646195
29. Kriegel, H., Kröger, P., Schubert, E., Zimek, A.: Outlier detection in axis-parallel subspaces of high dimensional data. In: PAKDD. Lecture Notes in Computer Science, vol. 5476, pp. 831–838. Springer (2009). https://doi.org/10.1007/978-3-642-01307-2_86
30. Kriegel, H.P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 08. pp. 444–452. ACM Press, New York, NY, USA (2008). https://doi.org/10.1145/1401890.1401946
31. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1106–1114 (2012). https://doi.org/10.1145/3065386
32. Linnainmaa, S.: Taylor expansion of the accumulated rounding error. BIT Numerical Mathematics **16**(2), 146–160 (1976). https://doi.org/10.1007/BF01931367
33. Liu, F.T., Ting, K.M., Zhou, Z.: Isolation forest. In: ICDM. pp. 413–422. IEEE Computer Society (2008). https://doi.org/10.1109/ICDM.2008.17
34. Marchi, E., Vesperini, F., Eyben, F., Squartini, S., Schuller, B.W.: A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks. In: ICASSP. pp. 1996–2000. IEEE (2015). https://doi.org/10.1109/ICASSP.2015.7178320
35. Müller, E., Schiffer, M., Seidl, T.: Adaptive outlierness for subspace outlier ranking. In: CIKM. pp. 1629–1632. ACM (2010). https://doi.org/10.1145/1871437.1871690
36. Müller, E., Schiffer, M., Seidl, T.: Statistical selection of relevant subspace projections for outlier ranking. In: ICDE. pp. 434–445. IEEE Computer Society (2011). https://doi.org/10.1109/ICDE.2011.5767916
37. Muñoz, A., Muruzábal, J.: Self-organising maps for outlier detection. Neurocomputing **18**(1), 33–60 (1998). https://doi.org/10.1016/S0925-2312(97)00068-4
38. Nguyen, H.V., Gopalkrishnan, V., Assent, I.: An unbiased distance-based outlier detection approach for high-dimensional data. In: DASFAA (1). Lecture Notes in Computer Science, vol. 6587, pp. 138–152. Springer (2011). https://doi.org/10.1007/978-3-642-20149-3_12
39. Nguyen, H.V., Müller, E., Vreeken, J., Keller, F., Böhm, K.: CMI: An information-theoretic contrast measure for enhancing subspace cluster and outlier detection. SDM pp. 198–206 (2013). https://doi.org/10.1137/1.9781611972832.22
40. Provost, F.J., Fawcett, T.: Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In: KDD. pp. 43–48. AAAI Press (1997), `http://www.aaai.org/Library/KDD/1997/kdd97-007.php`
41. Ramaswamy, S., Rastogi, R., Shim, K.: Efficient algorithms for mining outliers from large data sets. In: SIGMOD Conference. pp. 427–438. ACM (2000). https://doi.org/10.1145/342009.335437
42. Rayana, S.: ODDS library (2016), `http://odds.cs.stonybrook.edu`
43. Reddy, K.K., Sarkar, S., Venugopalan, V., Giering, M.: Anomaly detection and fault disambiguation in large flight data: A multi-modal deep autoencoder approach. In: Proceedings of the Annual Conference of the Prognostics and Health Management Society, Denver, Colorado. PHMC'16, vol. 7, pp. 192–199. PHM Society, Rochester, NY, USA (2016), `http://www.phmsociety.org/node/2088/`

44. Rubinstein, R.: The cross-entropy method for combinatorial and continuous optimization. Methodology and Computing in Applied Probability **1**(2), 127–190 (1999). https://doi.org/10.1023/A:1010091220143
45. Sammon, J.W.: A nonlinear mapping for data structure analysis. IEEE Trans. Computers **18**(5), 401–409 (1969). https://doi.org/10.1109/T-C.1969.222678
46. Sathe, S., Aggarwal, C.C.: LODES: local density meets spectral outlier detection. In: SDM. pp. 171–179. SIAM (2016). https://doi.org/10.1137/1.9781611974348.20
47. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Computation **13**(7), 1443–1471 (2001). https://doi.org/10.1162/089976601750264965
48. Schubert, E., Koos, A., Emrich, T., Züfle, A., Schmid, K.A., Zimek, A.: A framework for clustering uncertain data. PVLDB **8**(12), 1976–1979 (2015), `http://www.vldb.org/pvldb/vol8/p1976-schubert.pdf`
49. Smolensky, P.: Information processing in dynamical systems: Foundations of harmony theory. In: Rumelhart, D.E., McClelland, J.L., PDP Research Group, C. (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1, pp. 194–281. MIT Press, Cambridge, MA, USA (1986), `http://dl.acm.org/citation.cfm?id=104279.104290`
50. Wittek, P.: Somoclu: An efficient distributed library for self-organizing maps. CoRR **abs/1305.1422** (2013), `http://arxiv.org/abs/1305.1422`
51. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. CoRR **abs/1212.5701** (2012), `http://arxiv.org/abs/1212.5701`