
Deep One-Class Classification

Lukas Ruff^{*1} Robert A. Vandermeulen^{*2} Nico Görnitz³ Lucas Deecke⁴ Shoaib A. Siddiqui^{2,5}
Alexander Binder⁶ Emmanuel Müller¹ Marius Kloft²

Abstract

Despite the great advances made by deep learning in many machine learning problems, there is a relative dearth of deep learning approaches for anomaly detection. Those approaches which do exist involve networks trained to perform a task other than anomaly detection, namely generative models or compression, which are in turn adapted for use in anomaly detection; they are not trained on an anomaly detection based objective. In this paper we introduce a new anomaly detection method—Deep Support Vector Data Description—, which is trained on an anomaly detection based objective. The adaptation to the deep regime necessitates that our neural network and training procedure satisfy certain properties, which we demonstrate theoretically. We show the effectiveness of our method on MNIST and CIFAR-10 image benchmark datasets as well as on the detection of adversarial examples of GT-SRB stop signs.

1. Introduction

Anomaly detection (AD) (Chandola et al., 2009; Aggarwal, 2016) is the task of discerning unusual samples in data. Typically, this is treated as an unsupervised learning problem where the anomalous samples are not known a priori and

^{*}Equal contribution

A part of the work was done while LR, RV, LD, and MK were with Department of Computer Science, Humboldt University of Berlin, Germany. ¹Hasso Plattner Institute, Potsdam, Germany

²Department of Computer Science, TU Kaiserslautern, Kaiserslautern, Germany ³Machine Learning Group, Department of Electrical Engineering & Computer Science, TU Berlin, Berlin, Germany ⁴School of Informatics, University of Edinburgh, Edinburgh, Scotland ⁵German Research Center for Artificial Intelligence (DFKI GmbH), Kaiserslautern, Germany ⁶ISTD pillar, Singapore University of Technology and Design, Singapore. Correspondence to: Lukas Ruff <contact@lukasruff.com>.

Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

it is assumed that the majority of the training dataset consists of “normal” data (here and elsewhere the term “normal” means *not anomalous* and is unrelated to the Gaussian distribution). The aim then is to learn a model that accurately describes “normality.” Deviations from this description are then deemed to be anomalies. This is also known as *one-class classification* (Moya et al., 1993). AD algorithms are often trained on data collected during the normal operating state of a machine or system for monitoring (Lavin & Ahmad, 2015). Other domains include intrusion detection for cybersecurity (Garcia-Teodoro et al., 2009), fraud detection (Phua et al., 2005), and medical diagnosis (Salem et al., 2013; Schlegl et al., 2017). As with many fields, the data in these domains is growing rapidly in size and dimensionality and thus we require effective and efficient ways to detect anomalies in large quantities of high-dimensional data.

Classical AD methods such as the One-Class SVM (OC-SVM) (Schölkopf et al., 2001) or Kernel Density Estimation (KDE) (Parzen, 1962), often fail in high-dimensional, data-rich scenarios due to bad computational scalability and the curse of dimensionality. To be effective, such shallow methods typically require substantial feature engineering. In comparison, *deep learning* (LeCun et al., 2015; Schmidhuber, 2015) presents a way to learn relevant features automatically, with exceptional successes over classical methods (Collobert et al., 2011; Hinton et al., 2012), especially in computer vision (Krizhevsky et al., 2012; He et al., 2016). How to transfer the benefits of deep learning to AD is less clear, however, since finding the right unsupervised deep objective is hard (Bengio et al., 2013). Current approaches to deep AD have shown promising results (Hawkins et al., 2002; Sakurada & Yairi, 2014; Xu et al., 2015; Erfani et al., 2016; Andrews et al., 2016; Chen et al., 2017), but none of these methods are trained by optimizing an AD based objective function and typically rely on reconstruction error based heuristics.

In this work we introduce a novel approach to deep AD inspired by kernel-based one-class classification and minimum volume estimation. Our method, *Deep Support Vector Data Description (Deep SVDD)*, trains a neural network while minimizing the volume of a hypersphere that encloses the network representations of the data (see Figure 1). Minimizing the volume of the hypersphere forces the network to

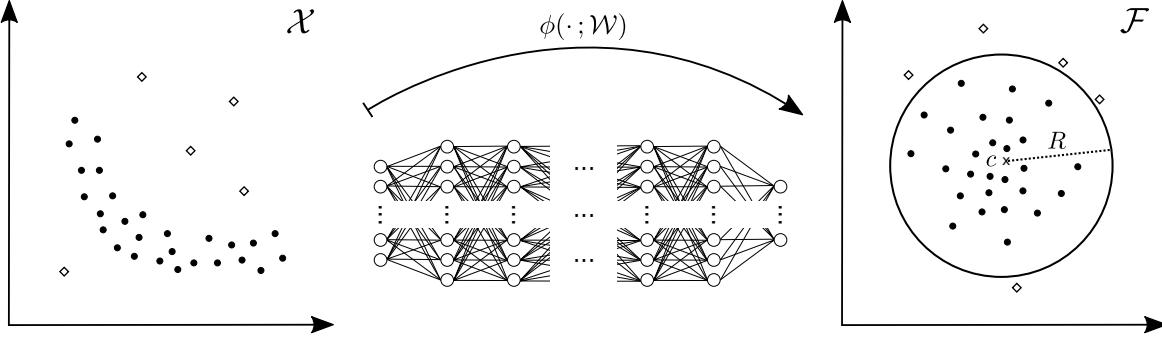


Figure 1. Deep SVDD learns a neural network transformation $\phi(\cdot; \mathcal{W})$ with weights \mathcal{W} from input space $\mathcal{X} \subseteq \mathbb{R}^d$ to output space $\mathcal{F} \subseteq \mathbb{R}^p$ that attempts to map most of the data network representations into a hypersphere characterized by center c and radius R of minimum volume. Mappings of normal examples fall within, whereas mappings of anomalies fall outside the hypersphere.

extract the common factors of variation since the network must closely map the data points to the center of the sphere.

2. Related Work

Before introducing Deep SVDD we briefly review kernel-based one-class classification and present existing deep approaches to AD.

2.1. Kernel-based One-Class Classification

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the data space. Let $k : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty)$ be a PSD kernel, \mathcal{F}_k its associated RKHS, and $\phi_k : \mathcal{X} \rightarrow \mathcal{F}_k$ its associated feature mapping. So $k(\mathbf{x}, \tilde{\mathbf{x}}) = \langle \phi_k(\mathbf{x}), \phi_k(\tilde{\mathbf{x}}) \rangle_{\mathcal{F}_k}$ for all $\mathbf{x}, \tilde{\mathbf{x}} \in \mathcal{X}$ where $\langle \cdot, \cdot \rangle_{\mathcal{F}_k}$ is the dot product in Hilbert space \mathcal{F}_k (Aronszajn, 1950). We review two kernel machine approaches to AD.

Probably the most prominent example of a kernel-based method for one-class classification is the One-Class SVM (OC-SVM) (Schölkopf et al., 2001). The objective of the OC-SVM finds a maximum margin hyperplane in feature space, $\mathbf{w} \in \mathcal{F}_k$, that best separates the mapped data from the origin. Given a dataset $\mathcal{D}_n = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ with $\mathbf{x}_i \in \mathcal{X}$, the OC-SVM solves the primal problem

$$\begin{aligned} \min_{\mathbf{w}, \rho, \xi} \quad & \frac{1}{2} \|\mathbf{w}\|_{\mathcal{F}_k}^2 - \rho + \frac{1}{\nu n} \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \langle \mathbf{w}, \phi_k(\mathbf{x}_i) \rangle_{\mathcal{F}_k} \geq \rho - \xi_i, \quad \xi_i \geq 0, \quad \forall i. \end{aligned} \quad (1)$$

Here ρ is the distance from the origin to hyperplane \mathbf{w} . Nonnegative slack variables $\xi = (\xi_1, \dots, \xi_n)^\top$ allow the margin to be soft, but violations ξ_i get penalized. $\|\mathbf{w}\|_{\mathcal{F}_k}^2$ is a regularizer on the hyperplane \mathbf{w} where $\|\cdot\|_{\mathcal{F}_k}$ is the norm induced by $\langle \cdot, \cdot \rangle_{\mathcal{F}_k}$. The hyperparameter $\nu \in (0, 1]$ controls the trade-off in the objective. Separating the data from the origin in feature space translates into finding a halfspace in which most of the data lie and points lying outside this halfspace, i.e. $\langle \mathbf{w}, \phi_k(\mathbf{x}) \rangle_{\mathcal{F}_k} < \rho$, are deemed

to be anomalous.

Support Vector Data Description (SVDD) (Tax & Duin, 2004) is a technique related to OC-SVM where a hypersphere is used to separate the data instead of a hyperplane. The objective of SVDD is to find the smallest hypersphere with center $\mathbf{c} \in \mathcal{F}_k$ and radius $R > 0$ that encloses the majority of the data in feature space \mathcal{F}_k . The SVDD primal problem is given by

$$\begin{aligned} \min_{R, \mathbf{c}, \xi} \quad & R^2 + \frac{1}{\nu n} \sum_i \xi_i \\ \text{s.t.} \quad & \|\phi_k(\mathbf{x}_i) - \mathbf{c}\|_{\mathcal{F}_k}^2 \leq R^2 + \xi_i, \quad \xi_i \geq 0, \quad \forall i. \end{aligned} \quad (2)$$

Again, slack variables $\xi_i \geq 0$ allow a soft boundary and hyperparameter $\nu \in (0, 1]$ controls the trade-off between penalties ξ_i and the volume of the sphere. Points which fall outside the sphere, i.e. $\|\phi_k(\mathbf{x}) - \mathbf{c}\|_{\mathcal{F}_k}^2 > R^2$, are deemed anomalous.

The OC-SVM and SVDD are closely related. Both methods can be solved by their respective duals, which are quadratic programs and can be solved via a variety of methods, e.g. *sequential minimal optimization* (Platt, 1998). In the case of the widely used Gaussian kernel, the two methods are equivalent and are asymptotically consistent density level set estimators (Tsybakov, 1997; Vert & Vert, 2006). Formulating the primal problems with hyperparameter $\nu \in (0, 1]$ as in (1) and (2) is a handy choice of parameterization since $\nu \in (0, 1]$ is (i) an upper bound on the fraction of outliers, and (ii) a lower bound on the fraction of support vectors (points that are either on or outside the boundary). This result is known as the ν -property (Schölkopf et al., 2001) and allows one to incorporate a prior belief about the fraction of outliers present in the training data into the model.

Apart from the necessity to perform explicit feature engineering (Pal & Foody, 2010), another drawback of the aforementioned methods is their poor computational scaling due to the construction and manipulation of the kernel

matrix. Kernel-based methods scale at least quadratically in the number of samples (Vempati et al., 2010) unless some sort of approximation technique is used (Rahimi & Recht, 2007). Moreover, prediction with kernel methods requires storing support vectors which can require large amounts of memory. As we will see, Deep SVDD does not suffer from these limitations.

2.2. Deep Approaches to Anomaly Detection

Deep learning (LeCun et al., 2015; Schmidhuber, 2015) is a subfield of *representation learning* (Bengio et al., 2013) that utilizes model architectures with multiple processing layers to learn data representations with multiple levels of abstraction. Multiple levels of abstraction allow for the representation of a rich space of features in a very compact and distributed form. Deep (multi-layered) neural networks are especially well-suited for learning representations of data that are hierarchical in nature, such as images or text.

We categorize approaches that try to leverage deep learning for AD into either “mixed” or “fully deep.” In mixed approaches, representations are learned separately in a preceding step before these representations are then fed into classical (shallow) AD methods like the OC-SVM. Fully deep approaches, in contrast, employ the representation learning objective directly for detecting anomalies.

With Deep SVDD, we introduce a novel, fully deep approach to unsupervised AD. Deep SVDD learns to extract the common factors of variation of the data distribution by training a neural network to fit the network outputs into a hypersphere of minimum volume. In comparison, virtually all existing deep AD approaches rely on the reconstruction error — either in mixed approaches for just learning representations, or directly for both representation learning as well as detection.

Deep autoencoders (Hinton & Salakhutdinov, 2006) (of various types) are the predominant approach used for deep AD. Autoencoders are neural networks which attempt to learn the identity function while having an intermediate representation of reduced dimension (or some sparsity regularization) serving as a bottleneck to induce the network to extract salient features from some dataset. Typically these networks are trained to minimize reconstruction error, i.e. $\|\mathbf{x} - \hat{\mathbf{x}}\|^2$. Therefore these networks should be able to extract the common factors of variation from normal samples and reconstruct them accurately, while anomalous samples do not contain these common factors of variation and thus cannot be reconstructed accurately. This allows for the use of autoencoders in mixed approaches (Xu et al., 2015; Andrews et al., 2016; Erfani et al., 2016; Sabokrou et al., 2016), by plugging the learned embeddings into classical AD methods, but also in fully deep approaches, by directly employing the reconstruction error as an anomaly score (Hawkins et al.,

2002; Sakurada & Yairi, 2014; An & Cho, 2015; Chen et al., 2017). Some variants of the autoencoder used for the purpose of AD include denoising autoencoders (Vincent et al., 2008; 2010), sparse autoencoders (Makhzani & Frey, 2013), variational autoencoders (VAEs) (Kingma & Welling, 2013), and deep convolutional autoencoders (DCAEs) (Masci et al., 2011; Makhzani & Frey, 2015), where the last variant is predominantly used in AD applications with image or video data (Seeböck et al., 2016; Richter & Roy, 2017).

Autoencoders have the objective of dimensionality reduction and do not target AD directly. The main difficulty of applying autoencoders for AD is given in choosing the right degree of compression, i.e. dimensionality reduction. If there was no compression, an autoencoder would just learn the identity function. In the other edge case of information reduction to a single value, the mean would be the optimal solution. That is, the “compactness” of the data representation is a model hyperparameter and choosing the right balance is hard due to the unsupervised nature and since the intrinsic dimensionality of the data is often difficult to estimate (Bengio et al., 2013). In comparison, we include the compactness of representation into our Deep SVDD objective by minimizing the volume of a data-enclosing hypersphere and thus target AD directly.

Apart from autoencoders, Schlegl et al. (2017) have recently proposed a novel deep AD method based on Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) called *AnoGAN*. In this method one first trains a GAN to generate samples according to the training data. Given a test point AnoGAN tries to find the point in the generator’s latent space that generates the sample closest to the test input considered. Intuitively, if the GAN has captured the distribution of the training data then normal samples, i.e. samples from the distribution, should have a good representation in the latent space and anomalous samples will not. To find the point in latent space, Schlegl et al. (2017) perform gradient descent in latent space keeping the learned weights of the generator fixed. AnoGAN finally defines an anomaly score also via the reconstruction error. Similar to autoencoders, a main difficulty of this generative approach is the question of how to regularize the generator for compactness.

3. Deep SVDD

In this section we introduce Deep SVDD, a method for deep one-class classification. We present the Deep SVDD objective, its optimization, and theoretical properties.

3.1. The Deep SVDD Objective

With Deep SVDD, we build on the kernel-based SVDD and minimum volume estimation by finding a data-enclosing hypersphere of smallest size. However, with Deep SVDD

we learn useful feature representations of the data together with the one-class classification objective. To do this we employ a neural network that is jointly trained to map the data into a hypersphere of minimum volume.

For some input space $\mathcal{X} \subseteq \mathbb{R}^d$ and output space $\mathcal{F} \subseteq \mathbb{R}^p$, let $\phi(\cdot; \mathcal{W}) : \mathcal{X} \rightarrow \mathcal{F}$ be a neural network with $L \in \mathbb{N}$ hidden layers and set of weights $\mathcal{W} = \{\mathbf{W}^1, \dots, \mathbf{W}^L\}$ where \mathbf{W}^ℓ are the weights of layer $\ell \in \{1, \dots, L\}$. That is, $\phi(\mathbf{x}; \mathcal{W}) \in \mathcal{F}$ is the feature representation of $\mathbf{x} \in \mathcal{X}$ given by network ϕ with parameters \mathcal{W} . The aim of Deep SVDD then is to jointly learn the network parameters \mathcal{W} together with minimizing the volume of a data-enclosing hypersphere in output space \mathcal{F} that is characterized by radius $R > 0$ and center $\mathbf{c} \in \mathcal{F}$ which we assume to be given for now. Given some training data $\mathcal{D}_n = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ on \mathcal{X} , we define the *soft-boundary Deep SVDD* objective as

$$\begin{aligned} \min_{R, \mathcal{W}} \quad & R^2 + \frac{1}{\nu n} \sum_{i=1}^n \max\{0, \|\phi(\mathbf{x}_i; \mathcal{W}) - \mathbf{c}\|^2 - R^2\} \\ & + \frac{\lambda}{2} \sum_{\ell=1}^L \|\mathbf{W}^\ell\|_F^2. \end{aligned} \quad (3)$$

As in kernel SVDD, minimizing R^2 minimizes the volume of the hypersphere. The second term is a penalty term for points lying outside the sphere after being passed through the network, i.e. if its distance to the center $\|\phi(\mathbf{x}_i; \mathcal{W}) - \mathbf{c}\|$ is greater than radius R . Hyperparameter $\nu \in (0, 1]$ controls the trade-off between the volume of the sphere and violations of the boundary, i.e. allowing some points to be mapped outside the sphere. We prove in Section 3.3 that the ν -parameter in fact allows us to control the proportion of outliers in a model similar to the ν -property of kernel methods mentioned previously. The last term is a weight decay regularizer on the network parameters \mathcal{W} with hyperparameter $\lambda > 0$, where $\|\cdot\|_F$ denotes the Frobenius norm.

Optimizing objective (3) lets the network learn parameters \mathcal{W} such that data points are closely mapped to the center \mathbf{c} of the hypersphere. To achieve this the network must extract the common factors of variation of the data. As a result, normal examples of the data are closely mapped to center \mathbf{c} , whereas anomalous examples are mapped further away from the center or outside of the hypersphere. Through this we obtain a compact description of the normal class. Minimizing the size of the sphere enforces this learning process.

For the case where we assume most of the training data \mathcal{D}_n is normal, which is often the case in one-class classification tasks, we propose an additional simplified objective. We

define the *One-Class Deep SVDD* objective as

$$\min_{\mathcal{W}} \quad \frac{1}{n} \sum_{i=1}^n \|\phi(\mathbf{x}_i; \mathcal{W}) - \mathbf{c}\|^2 + \frac{\lambda}{2} \sum_{\ell=1}^L \|\mathbf{W}^\ell\|_F^2. \quad (4)$$

One-Class Deep SVDD simply employs a quadratic loss for penalizing the distance of every network representation $\phi(\mathbf{x}_i; \mathcal{W})$ to $\mathbf{c} \in \mathcal{F}$. The second term again is a network weight decay regularizer with hyperparameter $\lambda > 0$. We can think of One-Class Deep SVDD also as finding a hypersphere of minimum volume with center \mathbf{c} . But unlike in soft-boundary Deep SVDD, where the hypersphere is contracted by penalizing the radius directly and the data representations that fall outside the sphere, One-Class Deep SVDD contracts the sphere by minimizing the mean distance of all data representations to the center. Again, to map the data (on average) as close to center \mathbf{c} as possible, the neural network must extract the common factors of variation. Penalizing the mean distance over all data points instead of allowing some points to fall outside the hypersphere is consistent with the assumption that the majority of training data is from one class.

For a given test point $\mathbf{x} \in \mathcal{X}$, we can naturally define an anomaly score s for both variants of Deep SVDD by the distance of the point to the center of the hypersphere, i.e.

$$s(\mathbf{x}) = \|\phi(\mathbf{x}; \mathcal{W}^*) - \mathbf{c}\|^2, \quad (5)$$

where \mathcal{W}^* are the network parameters of a trained model. For soft-boundary Deep SVDD, we can adjust this score by subtracting the final radius R^* of the trained model such that anomalies (points with representations outside the sphere) have positive scores, whereas inliers have negative scores. Note that the network parameters \mathcal{W}^* (and R^*) completely characterize a Deep SVDD model and no data must be stored for prediction, thus endowing Deep SVDD a very low memory complexity. This also allows fast testing by simply evaluating the network ϕ with learned parameters \mathcal{W}^* at some test point $\mathbf{x} \in \mathcal{X}$ which usually is just a concatenation of simple functions.

We address Deep SVDD optimization and selection of the hypersphere center $\mathbf{c} \in \mathcal{F}$ in the following two subsections.

3.2. Optimization of Deep SVDD

We use stochastic gradient descent (SGD) and its variants (e.g., Adam (Kingma & Ba, 2014)) to optimize the parameters \mathcal{W} of the neural network in both Deep SVDD objectives using backpropagation. Training is carried out until convergence to a local minimum. Using SGD allows Deep SVDD to scale well with large datasets as its computational complexity scales linearly in the number of training batches and each batch can be processed in parallel (e.g. by processing on multiple GPUs). SGD optimization also enables iterative or online learning.

Since the network parameters \mathcal{W} and radius R generally live on different scales, using one common SGD learning rate may be inefficient for optimizing the soft-boundary Deep SVDD. Instead, we suggest optimizing the network parameters \mathcal{W} and radius R alternately in an alternating minimization/block coordinate descent approach. That is, we train the network parameters \mathcal{W} for some $k \in \mathbb{N}$ epochs while the radius R is fixed. Then, after every k -th epoch, we solve for radius R given the data representations from the network using the network parameters \mathcal{W} of the latest update. R can be easily solved for via line search.

3.3. Properties of Deep SVDD

For an improperly formulated network or hypersphere center c , the Deep SVDD can learn trivial, uninformative solutions. Here we theoretically demonstrate some network properties which will yield trivial solutions (and thus must be avoided). We then prove the ν -property for soft-boundary Deep SVDD.

In the following let $J_{\text{soft}}(\mathcal{W}, R)$ and $J_{\text{OC}}(\mathcal{W})$ be the soft-boundary and One-Class Deep SVDD objective functions as defined in (3) and (4). First, we show that including the hypersphere center $c \in \mathcal{F}$ as a free optimization variable leads to trivial solutions for both objectives.

Proposition 1 (All-zero-weights solution). *Let \mathcal{W}_0 be the set of all-zero network weights, i.e., $\mathbf{W}^\ell = \mathbf{0}$ for every $\mathbf{W}^\ell \in \mathcal{W}_0$. For this choice of parameters, the network maps any input to the same output, i.e., $\phi(\mathbf{x}; \mathcal{W}_0) = \phi(\tilde{\mathbf{x}}; \mathcal{W}_0) =: c_0 \in \mathcal{F}$ for any $\mathbf{x}, \tilde{\mathbf{x}} \in \mathcal{X}$. Then, if $c = c_0$, the optimal solution of Deep SVDD is given by $\mathcal{W}^* = \mathcal{W}_0$ and $R^* = 0$.*

Proof. For every configuration (\mathcal{W}, R) we have that $J_{\text{soft}}(R, \mathcal{W}) \geq 0$ and $J_{\text{OC}}(\mathcal{W}) \geq 0$ respectively. As the output of the all-zero-weights network $\phi(\mathbf{x}; \mathcal{W}_0)$ is constant for every input $\mathbf{x} \in \mathcal{X}$ (all parameters in each network unit are zero and thus the linear projection in each network unit maps any input to zero), and the center of the hypersphere is given by $c = \phi(\mathbf{x}; \mathcal{W}_0)$, all errors in the empirical sums of the objectives become zero. Thus, $R^* = 0$ and $\mathcal{W}^* = \mathcal{W}_0$ are optimal solutions since $J_{\text{soft}}(\mathcal{W}^*, R^*) = 0$ and $J_{\text{OC}}(\mathcal{W}^*) = 0$ in this case. \square

Stated less formally, Proposition 1 implies that if we include the hypersphere center c as a free variable in the SGD optimization, Deep SVDD would likely converge to the trivial solution $(\mathcal{W}^*, R^*, c^*) = (\mathcal{W}_0, 0, c_0)$. We call such a solution, where the network learns weights such that the network produces a constant function mapping to the hypersphere center, ‘‘hypersphere collapse’’ since the hypersphere radius collapses to zero. Proposition 1 also implies that we require $c \neq c_0$ when fixing c in output space \mathcal{F} because otherwise a hypersphere collapse would again be possible. For a

convolutional neural network (CNN) with ReLU activation functions, for example, this would require $c \neq 0$. We found empirically that fixing c as the mean of the network representations that result from performing an initial forward pass on some training data sample to be a good strategy. Although we obtained similar results in our experiments for other choices of c (making sure $c \neq c_0$), we found that fixing c in the neighborhood of the initial network outputs made SGD convergence faster and more robust.

Next, we identify two network architecture properties, that would also enable trivial hypersphere collapse solutions.

Proposition 2 (Bias terms). *Let $c \in \mathcal{F}$ be any fixed hypersphere center. If there is any hidden layer in network $\phi(\cdot; \mathcal{W}) : \mathcal{X} \rightarrow \mathcal{F}$ having a bias term, there exists an optimal solution (R^*, \mathcal{W}^*) of the Deep SVDD objectives (3) and (4) with $R^* = 0$ and $\phi(\mathbf{x}; \mathcal{W}^*) = c$ for every $\mathbf{x} \in \mathcal{X}$.*

Proof. Assume layer $\ell \in \{1, \dots, L\}$ with weights \mathbf{W}^ℓ also has a bias term \mathbf{b}^ℓ . For any input $\mathbf{x} \in \mathcal{X}$, the output of layer ℓ is then given by

$$\mathbf{z}^\ell(\mathbf{x}) = \sigma^\ell(\mathbf{W}^\ell \cdot \mathbf{z}^{\ell-1}(\mathbf{x}) + \mathbf{b}^\ell),$$

where ‘‘.’’ denotes a linear operator (e.g., matrix multiplication or convolution), $\sigma^\ell(\cdot)$ is the activation of layer ℓ , and the output $\mathbf{z}^{\ell-1}$ of the previous layer $\ell - 1$ depends on input \mathbf{x} by the concatenation of previous layers. Then, for $\mathbf{W}^\ell = \mathbf{0}$, we have that $\mathbf{z}^\ell(\mathbf{x}) = \sigma^\ell(\mathbf{b}^\ell)$, i.e., the output of layer ℓ is constant for every input $\mathbf{x} \in \mathcal{X}$. Therefore, the bias term \mathbf{b}^ℓ (and the weights of the subsequent layers) can be chosen such that $\phi(\mathbf{x}; \mathcal{W}^*) = c$ for every $\mathbf{x} \in \mathcal{X}$ (assuming c is in the image of the network as a function of \mathbf{b}^ℓ and the subsequent parameters $\mathbf{W}^{\ell+1}, \dots, \mathbf{W}^L$). Hence, selecting \mathcal{W}^* in this way results in an empirical term of zero and choosing $R^* = 0$ gives the optimal solution (ignoring the weight decay regularization terms for simplicity). \square

Put differently, Proposition 2 implies that networks with bias terms can easily learn any constant function, which is independent of the input $\mathbf{x} \in \mathcal{X}$. It follows that bias terms should not be used in neural networks with Deep SVDD since the network can learn the constant function mapping directly to the hypersphere center, leading to hypersphere collapse.¹

Proposition 3 (Bounded activation functions). *Consider a network unit having a monotonic activation function $\sigma(\cdot)$ that has an upper (or lower) bound with $\sup_z \sigma(z) \neq 0$ (or $\inf_z \sigma(z) \neq 0$). Then, for a set of unit inputs $\{\mathbf{z}_1, \dots, \mathbf{z}_n\}$ that have at least one feature that is positive or negative for all inputs, the non-zero supremum (or infimum) can be uniformly approximated on the set of inputs.*

¹Proposition 2 also explains why autoencoders with bias terms are vulnerable to converge to a constant mapping onto the mean, which is the optimal constant solution of the mean squared error.

Proof. W.l.o.g. consider the case of σ being upper bounded by $B := \sup_z \sigma(z) \neq 0$ and feature k being positive for all inputs, i.e. $z_i^{(k)} > 0$ for every $i = 1, \dots, n$. Then, for every $\varepsilon > 0$, one can always choose the weight of the k -th element w_k large enough (setting all other network unit weights to zero) such that $\sup_i |\sigma(w_k z_i^{(k)}) - B| < \varepsilon$. \square

Proposition 3 simply says that a network unit with bounded activation function can be saturated for all inputs having at least one feature with common sign thereby emulating a bias term in the subsequent layer, which again leads to a hypersphere collapse. Therefore, unbounded activation functions (or functions only bounded by 0) such as the ReLU should be preferred in Deep SVDD to avoid a hypersphere collapse due to “learned” bias terms.

To summarize the above analysis: the choice of hypersphere center c must be something other than the all-zero-weights solution and only neural networks without bias terms or bounded activation functions should be used in Deep SVDD to prevent a hypersphere collapse solution. Lastly, we prove that the ν -property also holds for soft-boundary Deep SVDD which allows to include a prior assumption on the number of anomalies assumed to be present in the training data.

Proposition 4 (ν -property). *Hyperparameter $\nu \in (0, 1]$ in the soft-boundary Deep SVDD objective in (3) is an upper bound on the fraction of outliers and a lower bound on the fraction of samples being outside or on the boundary of the hypersphere.*

Proof. Define $d_i = \|\phi(\mathbf{x}_i; \mathcal{W}) - c\|^2$ for $i = 1, \dots, n$. W.l.o.g. assume $d_1 \geq \dots \geq d_n$. The number of outliers is then given by $n_{\text{out}} = |\{i : d_i > R^2\}|$ and we can write the soft-boundary objective J_{soft} (in radius R) as

$$J_{\text{soft}}(R) = R^2 - \frac{n_{\text{out}}}{\nu n} R^2 = \left(1 - \frac{n_{\text{out}}}{\nu n}\right) R^2.$$

That is, radius R is decreased as long as $n_{\text{out}} \leq \nu n$ holds and decreasing R gradually increases n_{out} . Thus, $\frac{n_{\text{out}}}{n} \leq \nu$ must hold in the optimum, i.e. ν is an upper bound on the fraction of outliers, and the optimal radius R^* is given for the largest n_{out} for which this inequality still holds. Finally, we have that $R^{*2} = d_i$ for $i = n_{\text{out}} + 1$ since radius R is minimal in this case and points on the boundary do not increase the objective. Hence, we also have $|\{i : d_i \geq R^{*2}\}| \geq n_{\text{out}} + 1 \geq \nu n$. \square

4. Experiments

We evaluate Deep SVDD on the well-known MNIST (Le-Cun et al., 2010) and CIFAR-10 (Krizhevsky & Hinton, 2009) datasets. Adversarial attacks (Goodfellow et al., 2015) have seen a lot attention recently and here we examine the possibility of using anomaly detection to detect such attacks.

To do this we apply Boundary Attack (Brendel et al., 2018) to the GTSRB stop signs dataset (Stallkamp et al., 2011). We compare our method against a diverse collection of state-of-the-art methods from different paradigms. We use image data since they are usually high-dimensional and moreover allow for a qualitative visual assessment of detected anomalies by human observers. Using classification datasets to create one-class classification setups allows us to evaluate the results quantitatively via AUC measure by using the ground truth labels in testing (cf. Erfani et al., 2016; Em-mott et al., 2016). For training, of course, we do not use any labels.²

4.1. Competing methods

Shallow Baselines (i) Kernel OC-SVM/SVDD with Gaussian kernel. We select the inverse length scale γ from $\gamma \in \{2^{-10}, 2^{-9}, \dots, 2^{-1}\}$ via grid search using the performance on a small holdout set (10 % of randomly drawn test samples). This grants shallow SVDD a small supervised advantage. We run all experiments for $\nu \in \{0.01, 0.1\}$ and report the better result. (ii) Kernel density estimation (KDE). We select the bandwidth h of the Gaussian kernel from $h \in \{2^{0.5}, 2^1, \dots, 2^5\}$ via 5-fold cross-validation using the log-likelihood score. (iii) Isolation Forest (IF) (Liu et al., 2008). We set the number of trees to $t = 100$ and the sub-sampling size to $\psi = 256$, as recommended in the original work. We do not compare to lazy evaluation approaches since such methods have no training phase and do not learn a model of normality (e.g. Local Outlier Factor (LOF) (Breunig et al., 2000)). For all three shallow baselines, we reduce the dimensionality of the data via PCA, where we choose the minimum number of eigenvectors such that at least 95% of the variance is retained (cf. Erfani et al., 2016).

Deep Baselines and Deep SVDD We compare Deep SVDD to the two deep approaches described Section 2.2. We choose the DCAE from the various autoencoders since our experiments are on image data. For the DCAE encoder, we employ the same network architectures as we use for Deep SVDD. The decoder is then created symmetrically, where we substitute max-pooling with upsampling. We train the DCAE using the MSE loss. For AnoGAN we fix the architecture to DCGAN (Radford et al., 2015) and set the latent space dimensionality to 256, following Metz et al. (2017), and otherwise follow Schlegl et al. (2017). For Deep SVDD, we remove the bias terms in all network units to prevent a hypersphere collapse as explained in Section 3.3. In soft-boundary Deep SVDD, we solve for R via line search every $k = 5$ epochs. We choose ν from $\nu \in \{0.01, 0.1\}$ and again report the best results. As was described in Sec-

²We provide our code at <https://github.com/lukasruff/Deep-SVDD>.

Table 1. Average AUCs in % with StdDevs (over 10 seeds) per method and one-class experiment on MNIST and CIFAR-10.

NORMAL CLASS	OC-SVM/ SVDD	KDE	IF	DCAE	ANOGAN	SOFT-BOUND. DEEP SVDD	ONE-CLASS DEEP SVDD
0	98.6 \pm 0.0	97.1 \pm 0.0	98.0 \pm 0.3	97.6 \pm 0.7	96.6 \pm 1.3	97.8 \pm 0.7	98.0 \pm 0.7
1	99.5 \pm 0.0	98.9 \pm 0.0	97.3 \pm 0.4	98.3 \pm 0.6	99.2 \pm 0.6	99.6 \pm 0.1	99.7 \pm 0.1
2	82.5 \pm 0.1	79.0 \pm 0.0	88.6 \pm 0.5	85.4 \pm 2.4	85.0 \pm 2.9	89.5 \pm 1.2	91.7 \pm 0.8
3	88.1 \pm 0.0	86.2 \pm 0.0	89.9 \pm 0.4	86.7 \pm 0.9	88.7 \pm 2.1	90.3 \pm 2.1	91.9 \pm 1.5
4	94.9 \pm 0.0	87.9 \pm 0.0	92.7 \pm 0.6	86.5 \pm 2.0	89.4 \pm 1.3	93.8 \pm 1.5	94.9 \pm 0.8
5	77.1 \pm 0.0	73.8 \pm 0.0	85.5 \pm 0.8	78.2 \pm 2.7	88.3 \pm 2.9	85.8 \pm 2.5	88.5 \pm 0.9
6	96.5 \pm 0.0	87.6 \pm 0.0	95.6 \pm 0.3	94.6 \pm 0.5	94.7 \pm 2.7	98.0 \pm 0.4	98.3 \pm 0.5
7	93.7 \pm 0.0	91.4 \pm 0.0	92.0 \pm 0.4	92.3 \pm 1.0	93.5 \pm 1.8	92.7 \pm 1.4	94.6 \pm 0.9
8	88.9 \pm 0.0	79.2 \pm 0.0	89.9 \pm 0.4	86.5 \pm 1.6	84.9 \pm 2.1	92.9 \pm 1.4	93.9 \pm 1.6
9	93.1 \pm 0.0	88.2 \pm 0.0	93.5 \pm 0.3	90.4 \pm 1.8	92.4 \pm 1.1	94.9 \pm 0.6	96.5 \pm 0.3
AIRPLANE	61.6 \pm 0.9	61.2 \pm 0.0	60.1 \pm 0.7	59.1 \pm 5.1	67.1 \pm 2.5	61.7 \pm 4.2	61.7 \pm 4.1
AUTOMOBILE	63.8 \pm 0.6	64.0 \pm 0.0	50.8 \pm 0.6	57.4 \pm 2.9	54.7 \pm 3.4	64.8 \pm 1.4	65.9 \pm 2.1
BIRD	50.0 \pm 0.5	50.1 \pm 0.0	49.2 \pm 0.4	48.9 \pm 2.4	52.9 \pm 3.0	49.5 \pm 1.4	50.8 \pm 0.8
CAT	55.9 \pm 1.3	56.4 \pm 0.0	55.1 \pm 0.4	58.4 \pm 1.2	54.5 \pm 1.9	56.0 \pm 1.1	59.1 \pm 1.4
DEER	66.0 \pm 0.7	66.2 \pm 0.0	49.8 \pm 0.4	54.0 \pm 1.3	65.1 \pm 3.2	59.1 \pm 1.1	60.9 \pm 1.1
DOG	62.4 \pm 0.8	62.4 \pm 0.0	58.5 \pm 0.4	62.2 \pm 1.8	60.3 \pm 2.6	62.1 \pm 2.4	65.7 \pm 2.5
FROG	74.7 \pm 0.3	74.9 \pm 0.0	42.9 \pm 0.6	51.2 \pm 5.2	58.5 \pm 1.4	67.8 \pm 2.4	67.7 \pm 2.6
HORSE	62.6 \pm 0.6	62.6 \pm 0.0	55.1 \pm 0.7	58.6 \pm 2.9	62.5 \pm 0.8	65.2 \pm 1.0	67.3 \pm 0.9
SHIP	74.9 \pm 0.4	75.1 \pm 0.0	74.2 \pm 0.6	76.8 \pm 1.4	75.8 \pm 4.1	75.6 \pm 1.7	75.9 \pm 1.2
TRUCK	75.9 \pm 0.3	76.0 \pm 0.0	58.9 \pm 0.7	67.3 \pm 3.0	66.5 \pm 2.8	71.0 \pm 1.1	73.1 \pm 1.2

tion 3.3, we set the hypersphere center c to the mean of the mapped data after performing an initial forward pass. For optimization, we use the Adam optimizer (Kingma & Ba, 2014) with parameters as recommended in the original work and apply Batch Normalization (Ioffe & Szegedy, 2015). For the competing deep AD methods we initialize network weights by uniform Glorot weights (Glorot & Bengio, 2010), and for Deep SVDD use the weights from the trained DCAE encoder for initialization, thus establishing a pre-training procedure. We employ a simple two-phase learning rate schedule (searching + fine-tuning) with initial learning rate $\eta = 10^{-4}$, and subsequently $\eta = 10^{-5}$. For DCAE we train 250 + 100 epochs, for Deep SVDD 150 + 100. Leaky ReLU activations are used with leakiness $\alpha = 0.1$.

4.2. One-class classification on MNIST and CIFAR-10

Setup Both MNIST and CIFAR-10 have ten different classes from which we create ten one-class classification setups. In each setup, one of the classes is the normal class and samples from the remaining classes are used to represent anomalies. We use the original training and test splits in our experiments and only train with training set examples from the respective normal class. This gives training set sizes of $n \approx 6\,000$ for MNIST and $n = 5\,000$ for CIFAR-10. Both test sets have 10 000 samples including samples from the nine anomalous classes for each setup. We pre-process all images with global contrast normalization using the L^1 -norm and finally rescale to $[0, 1]$ via min-max-scaling.

Network architectures For both datasets, we use LeNet-type CNNs, where each convolutional module consists of a convolutional layer followed by leaky ReLU activations and 2×2 max-pooling. On MNIST, we use a CNN with two modules, $8 \times (5 \times 5 \times 1)$ -filters followed by $4 \times (5 \times 5 \times 1)$ -filters, and a final dense layer of 32 units. On CIFAR-10, we use a CNN with three modules, $32 \times (5 \times 5 \times 3)$ -filters, $64 \times (5 \times 5 \times 3)$ -filters, and $128 \times (5 \times 5 \times 3)$ -filters, followed by a final dense layer of 128 units. We use a batch size of 200 and set the weight decay hyperparameter to $\lambda = 10^{-6}$.

Results Results are presented in Table 1. Deep SVDD clearly outperforms both its shallow and deep competitors on MNIST. On CIFAR-10 the picture is mixed. Deep SVDD, however, shows an overall strong performance. It is interesting to note that shallow SVDD and KDE perform better than deep methods on three of the ten CIFAR-10 classes. Figures 2 and 3 show examples of the most normal and most anomalous in-class samples according to Deep SVDD and KDE respectively. We can see that normal examples of the classes on which KDE performs best seem to have strong global structures. For example, TRUCK images are mostly divided horizontally into street and sky, and DEER as well as FROG have similar colors globally. For these classes, choosing local CNN features can be questioned. These cases underline the importance of network architecture choice. Notably, the One-Class Deep SVDD performs slightly better than its soft-boundary counterpart on both datasets. This may be because the assumption of no anomalies being present in the training data is valid in our scenario. Due to SGD optimization, deep methods show higher standard deviations.

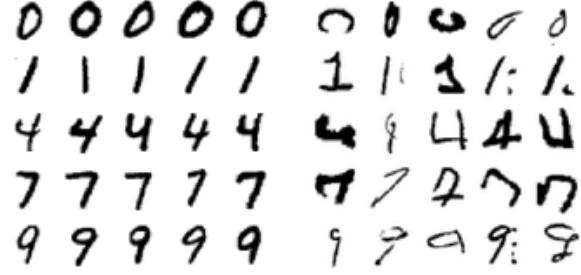


Figure 2. Most normal (left) and most anomalous (right) in-class examples determined by One-Class Deep SVDD for selected MNIST (top) and CIFAR-10 (bottom) one-class experiments.

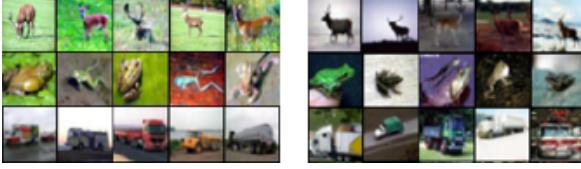


Figure 3. Most normal (left) and most anomalous (right) in-class examples determined by KDE for CIFAR-10 one-class experiments in which KDE performs best.

4.3. Adversarial attacks on GTSRB stop signs

Setup Detecting adversarial attacks is vital in many applications such autonomous driving. In this experiment, we test how Deep SVDD compares to its competitors on detecting adversarial examples. We consider the “stop sign” class of the German Traffic Sign Recognition Benchmark (GTSRB) dataset, for which we generate adversarial examples from randomly drawn stop sign images of the test set using Boundary Attack. We train the models again only on normal stop sign samples and in testing check if adversarial examples are correctly detected. The training set contains $n = 780$ stop signs. The test set is composed of 270 normal examples and 20 adversarial examples. We pre-process the data by removing the 10% border around each sign, and then resize every image to 32×32 pixels. After that, we again apply global contrast normalization using the L^1 -norm and rescale to the unit interval $[0, 1]$.

Table 2. Average AUCs in % with StdDevs (over 10 seeds) per method on GTSRB stop signs with adversarial attacks.

METHOD	AUC
OC-SVM/SVDD	67.5 ± 1.2
KDE	60.5 ± 1.7
IF	73.8 ± 0.9
DCAE	79.1 ± 3.0
ANOGAN	—
SOFT-BOUND. DEEP SVDD	77.8 ± 4.9
ONE-CLASS DEEP SVDD	80.3 ± 2.8

Network architecture We use a CNN with LeNet architecture having three convolutional modules, $16 \times (5 \times 5 \times 3)$ -filters, $32 \times (5 \times 5 \times 3)$ -filters, and $64 \times (5 \times 5 \times 3)$ -filters, followed by a final dense layer of 32 units. We train with a smaller batch size of 64, due to the dataset size and set again hyperparameter $\lambda = 10^{-6}$.

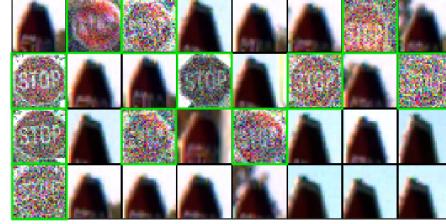


Figure 4. Most anomalous stop signs detected by One-Class Deep SVDD. Adversarial examples are highlighted in green.

Results Table 2 shows the results. The One-Class Deep SVDD shows again the best performance. Generally, the deep methods perform better. The DCGAN of AnoGAN did not converge due to the data set size which is too small for GANs. Figure 4 shows the most anomalous samples detected by One-Class Deep SVDD which are either adversarial attacks or images in odd perspectives that are cropped incorrectly. We refer to the supplementary material for more examples of the most normal images and anomalies detected.

5. Conclusion

We introduced the first fully deep one-class classification objective for unsupervised AD in this work. Our method, Deep SVDD, jointly trains a deep neural network while optimizing a data-enclosing hypersphere in output space. Through this Deep SVDD extracts common factors of variation from the data. We have demonstrated theoretical properties of our method such as the ν -property that allows to incorporate a prior assumption on the number of outliers being present in the data. Our experiments demonstrate quantitatively as well as qualitatively the sound performance of Deep SVDD.

Acknowledgements

We kindly thank the reviewers for their constructive feedback which helped to improve this work. LR acknowledges financial support from the German Federal Ministry of Transport and Digital Infrastructure (BMVI) in the project OSIMAB (FKZ: 19F2017E). AB is grateful for support by the SUTD startup grant and the STElectronics-SUTD Cybersecurity Laboratory. MK and RV acknowledge support from the German Research Foundation (DFG) award KL 2698/2-1 and from the Federal Ministry of Science and Education (BMBF) award 031B0187B.

References

- Aggarwal, C. *Outlier Analysis*. Springer, 2nd edition, 2016.
- An, J. and Cho, S. Variational Autoencoder based Anomaly Detection using Reconstruction Probability. *SNU Data Mining Center, Tech. Rep.*, 2015.
- Andrews, J. T. A., Morton, E. J., and Griffin, L. D. Detecting Anomalous Data Using Auto-Encoders. *IJMLC*, 6(1):21, 2016.
- Aronszajn, N. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.
- Bengio, Y., Courville, A., and Vincent, P. Representation Learning: A Review and New Perspectives. *IEEE TPAMI*, 35(8):1798–1828, 2013.
- Brendel, W., Rauber, J., and Bethge, M. Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models. In *ICLR*, 2018.
- Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. LOF: Identifying Density-Based Local Outliers. In *SIGMOD Record*, volume 29, pp. 93–104, 2000.
- Chandola, V., Banerjee, A., and Kumar, V. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(3):1–58, 2009.
- Chen, J., Sathe, S., Aggarwal, C., and Turaga, D. Outlier Detection with Autoencoder Ensembles. In *SDM*, pp. 90–98, 2017.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. Natural Language Processing (Almost) from Scratch. *JMLR*, 12(Aug):2493–2537, 2011.
- Emmott, A., Das, A., Dietterich, T., Fern, A., and Wong, W.-K. Anomaly detection meta-analysis benchmarks. 2016.
- Erfani, S. M., Rajasegarar, S., Karunasekera, S., and Leckie, C. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognition*, 58:121–134, 2016.
- Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., and Vázquez, E. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1-2):18–28, 2009.
- Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pp. 249–256, 2010.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. In *NIPS*, pp. 2672–2680, 2014.
- Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.
- Hawkins, S., He, H., Williams, G., and Baxter, R. Outlier Detection Using Replicator Neural Networks. In *DaWaK*, volume 2454, pp. 170–180, 2002.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *CVPR*, June 2016.
- Hinton, G. E. and Salakhutdinov, R. R. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313 (5786):504–507, 2006.
- Hinton, G. E., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. Deep Neural Networks for Acoustic Modeling in Speech Recognition. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.
- Ioffe, S. and Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *ICML*, pp. 448–456, 2015.
- Kingma, D. and Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*, 2014.
- Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *ICLR*, 2013.
- Krizhevsky, A. and Hinton, G. E. Learning multiple layers of features from tiny images. 2009.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*, pp. 1090–1098, 2012.
- Lavin, A. and Ahmad, S. Evaluating Real-time Anomaly Detection Algorithms — the Numenta Anomaly Benchmark. In *14th ICMLA*, pp. 38–44, 2015.

- LeCun, Y., Cortes, C., and Burges, C. MNIST handwritten digit database. *AT&T Labs*, 2, 2010.
- LeCun, Y., Bengio, Y., and Hinton, G. E. Deep Learning. *Nature*, 521(7553):436–444, 2015.
- Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation Forest. In *ICDM*, pp. 413–422, 2008.
- Makhzani, A. and Frey, B. K-sparse Autoencoders. *arXiv:1312.5663*, 2013.
- Makhzani, A. and Frey, B. J. Winner-Take-All Autoencoders. In *NIPS*, pp. 2791–2799, 2015.
- Masci, J., Meier, U., Cireşan, D., and Schmidhuber, J. Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction. *ICANN*, pp. 52–59, 2011.
- Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled Generative Adversarial Networks. In *ICLR*, 2017.
- Moya, M. M., Koch, M. W., and Hostetler, L. D. One-class classifier networks for target recognition applications. In *Proceedings World Congress on Neural Networks*, pp. 797–801, 1993.
- Pal, M. and Foody, G. M. Feature selection for classification of hyperspectral data by SVM. *IEEE Transactions on Geoscience and Remote Sensing*, 48(5):2297–2307, 2010.
- Parzen, E. On Estimation of a Probability Density Function and Mode. *The annals of mathematical statistics*, 33(3): 1065–1076, 1962.
- Phua, C., Lee, V., Smith, K., and Gayler, R. A Comprehensive Survey of Data Mining-based Fraud Detection Research. *Clayton School of Information Technology, Monash University, Tech. Rep.*, 2005.
- Platt, J. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.
- Radford, A., Metz, L., and Chintala, S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv:1511.06434*, 2015.
- Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *NIPS*, 2007.
- Richter, C. and Roy, N. Safe Visual Navigation via Deep Learning and Novelty Detection. In *Robotics: Science and Systems Conference*, 2017.
- Sabokrou, M., Fayyaz, M., Fathy, M., et al. Fully Convolutional Neural Network for Fast Anomaly Detection in Crowded Scenes. *arXiv:1609.00866*, 2016.
- Sakurada, M. and Yairi, T. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the 2nd MLSDA Workshop*, pp. 4, 2014.
- Salem, O., Guerassimov, A., Mehaoua, A., Marcus, A., and Furht, B. Sensor Fault and Patient Anomaly Detection and Classification in Medical Wireless Sensor Networks. In *ICC*, pp. 4373–4378, 2013.
- Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., and Langs, G. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In *IPMI*, pp. 146–157, 2017.
- Schmidhuber, J. Deep Learning in Neural Networks: An Overview. *Neural networks*, 61:85–117, 2015.
- Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. Estimating the Support of a High-Dimensional Distribution. *Neural computation*, 13(7): 1443–1471, 2001.
- Seeböck, P., Waldstein, S., Klimscha, S., Gerendas, B. S., Donner, R., Schlegl, T., Schmidt-Erfurth, U., and Langs, G. Identifying and Categorizing Anomalies in Retinal Imaging Data. *arXiv:1612.00686*, 2016.
- Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *IJCNN*, pp. 1453–1460, 2011.
- Tax, D. M. J. and Duin, R. P. W. Support Vector Data Description. *Machine learning*, 54(1):45–66, 2004.
- Tsybakov, A. B. On Nonparametric Estimation of Density Level Sets. *The Annals of Statistics*, 25(3):948–969, 1997.
- Vempati, S., Vedaldi, A., Zisserman, A., and Jawahar, C. Generalized RBF feature maps for Efficient Detection. In *21st BMVC*, pp. 1–11, 2010.
- Vert, R. and Vert, J.-P. Consistency and Convergence Rates of One-Class SVMs and Related Algorithms. *JMLR*, 7 (May):817–854, 2006.
- Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and Composing Robust Features with Denoising Autoencoders. In *ICML*, pp. 1096–1103, 2008.
- Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *JMLR*, 11(Dec):3371–3408, 2010.
- Xu, D., Ricci, E., Yan, Y., Song, J., and Sebe, N. Learning Deep Representations of Appearance and Motion for Anomalous Event Detection. In *BMVC*, pp. 8.1–8.12, 2015.