Machine Learning (2020) 109:1617–1633
https://doi.org/10.1007/s10994-020-05880-w

Anomaly detection with inexact labels

Tomoharu Iwata[1] · Machiko Toyoda[2] · Shotaro Tora[2] · Naonori Ueda[1]

Received: 18 September 2019 / Revised: 12 April 2020 / Accepted: 24 April 2020 / Published online: 31 May 2020

Abstract
We propose a supervised anomaly detection method for data with inexact anomaly labels, where each label, which is assigned to a set of instances, indicates that at least one instance in the set is anomalous. Although many anomaly detection methods have been proposed, they cannot handle inexact anomaly labels. To measure the performance with inexact anomaly labels, we define the inexact AUC, which is our extension of the area under the ROC curve (AUC) for inexact labels. The proposed method trains an anomaly score function so that the smooth approximation of the inexact AUC increases while anomaly scores for non-anomalous instances become low. We model the anomaly score function by a neural network-based unsupervised anomaly detection method, e.g., autoencoders. The proposed method performs well even when only a small number of inexact labels are available by incorporating an unsupervised anomaly detection mechanism with inexact AUC maximization. Using various datasets, we experimentally demonstrate that our proposed method improves the anomaly detection performance with inexact anomaly labels, and outperforms existing unsupervised and supervised anomaly detection and multiple instance learning methods.

Keywords Anomaly detection · Inexact labels · AUC maximization

1 Introduction

Anomaly detection is an important machine learning task, which is a task to find the anomalous instances in a dataset. Anomaly detection has been used in a wide variety of applications (Chandola et al. 2009; Patcha and Park 2007; Hodge and Austin 2004), such as network intrusion detection for cyber-security (Dokas et al. 2002; Yamanishi et al. 2004), fraud detection for credit cards (Aleskerov et al. 1997), defect detection in industrial machines (Fujimaki et al. 2005; Idé and Kashima 2004) and disease outbreak detection (Wong et al. 2003).

Editor: Jesse Davis.

✉ Tomoharu Iwata
tomoharu.iwata.gy@hco.ntt.co.jp

[1] NTT Communication Science Laboratories, Kyoto, Japan

[2] NTT Software Innovation Center, Tokyo, Japan

Many unsupervised anomaly detection methods have been proposed (Breunig et al. 2000; Schölkopf et al. 2001; Liu et al. 2008; Sakurada and Yairi 2014). When anomaly labels, which indicate whether each instance is anomalous, are given, the anomaly detection performance can be improved (Singh and Silakari 2009; Mukkamala et al. 2005; Rapaka et al. 2003; Nadeem et al. 2016; Gao et al. 2006; Das et al. 2016, 2017). However, it is difficult to attach exact anomaly labels in some situations. Consider one such example from server system failure detection, where server logs at each timestep is an instance, and we want to classify each instance into anomalous (system failure) or non-anomalous classes. System operators often do not know the exact timestep of failures; they only know that a failure occurred within a certain period of timesteps. In this case, anomaly labels are attached to multiple instances in the certain period of timesteps, in which non-anomalous instances might be included. Another example is detecting anomalous user behaviors in many computers, where each computer is simultaneously used by multiple users. Administrators can determine whether a computer has problems or not by checking its system logs, but they sometimes cannot identify which users are anomalous. In this case, anomaly labels are attached to multiple users who use the problematic computer.

In this paper, we propose a supervised anomaly detection method for data with inexact anomaly labels. An inexact anomaly label is attached to a set of instances, indicating that at least one instance in the set is anomalous. We call this set an inexact anomaly set. First, we define an extension of the area under the ROC curve (AUC) for performance measurement with inexact labels, which we call an *inexact AUC*. To the best of our knowledge, the inexact AUC is the first extension of the AUC that can handle inexact labels. Then we develop an anomaly detection method that maximizes the inexact AUC. With the proposed method, a function, which outputs an anomaly score given an instance, is modeled by the reconstruction error with autoencoders, which are a successfully used neural network-based unsupervised anomaly detection method (Sakurada and Yairi 2014; Sabokrou et al. 2016; Chong and Tay 2017; Zhou and Paffenroth 2017). Note that the proposed method can use any unsupervised anomaly detection methods with learnable parameters instead of autoencoders, such as variational autoencoders (Kingma and Wellniga 2014), energy-based models (Zhai et al. 2016), and isolation forests (Liu et al. 2008). The parameters of the anomaly score function are trained so that the anomaly scores for non-anomalous instances become low while the smooth approximation of the inexact AUC becomes high. Since our objective function is differentiable, the anomaly score function can be estimated efficiently using stochastic gradient-based optimization methods.

The proposed method performs well even when only a few inexact labels are given since it incorporates an unsupervised anomaly detection mechanism, which works without label information. In addition, the proposed method is robust to class imbalance since our proposed inexact AUC maximization is related to AUC maximization, which achieved high performance on imbalanced data classification tasks (Cortes and Mohri 2004). Class imbalance robustness is important for anomaly detection since anomalous instances occur more rarely than non-anomalous instances.

The remainder of the paper is organized as follows. In Sect. 2, we briefly review related work. In Sect. 3, we introduce AUC, which is the basis of the inexact AUC. In Sect. 4, we present the inexact AUC, define our task, and propose our method for supervised anomaly detection using inexact labels. In Sect. 5, we experimentally demonstrate the effectiveness of our proposed method using various datasets by comparing with existing anomaly detection and multiple instance learning methods. Finally, we present concluding remarks and discuss future work in Sect. 6.

2 Related work

Inexact labels in classification tasks have been considered in multiple instance learning methods (Dietterich et al. 1997; Maron and Lozano-Pérez 1998; Babenko et al. 2009; Wu et al. 2015; Cinbis et al. 2017). With standard supervised learning, we are given training instances that are individually labeled. On the other hand, with multiple instance learning, we are given labeled sets, where each set contains multiple instances. For a binary classification multiple instance learning task, a set is labeled negative if all the instances in it are negative, and it is labeled positive if it contains at least one positive instance. Inexact anomaly sets considered in this paper are the same with this definition on positive labeled sets, where a set is labeled anomalous if it contains at least one anomalous instance. Multiple instance learning has been used for many applications, such as drug activity prediction (Dietterich et al. 1997; Davis et al. 2007), image classification (Chen et al. 2006; Andrews et al. 2003; Zhang et al. 2002), and document classification (Zhou et al. 2009; Bunescu and Mooney 2007), but not for anomaly detection. With existing multiple instance learning methods, noisy-or (Maron and Lozano-Pérez 1998) and maximum (Andrews et al. 2003; Pinheiro and Collobert 2015; Zhu et al. 2017; Feng and Zhou 2017; Ilse et al. 2018) operators are used for modeling set-level labels from instance-lebel labels. The proposed method uses the maximum operator since anomaly scores are not probabilities, and noisy-or modeling requires probabilities. An advantage of the proposed method over existing multiple instance learning methods is that the proposed method works well with a small number of inexact anomaly labels using inexact AUC maximization and incorporating an unsupervised anomaly detection mechanism, where we exploit the characteristics of anomaly detection tasks. Existing multiple instance learning methods are not robust to class imbalance (Herrera et al. 2016; Carbonneau et al. 2018).

Anomaly detection is also called outlier detection (Hodge and Austin 2004) or novelty detection (Markou and Singh 2003). Aanomaly detecion has been used in a wide variety of applications (Chandola et al. 2009; Patcha and Park 2007; Hodge and Austin 2004), including network intrusion detection for cyber-security (Dokas et al. 2002; Yamanishi et al. 2004), fraud detection for credit cards (Aleskerov et al. 1997), defect detection in industrial machines (Fujimaki et al. 2005; Idé and Kashima 2004) and disease outbreak detection (Wong et al. 2003). Many unsupervised methods have been proposed, such as the local outlier factor (Breunig et al. 2000), one-class support vector machines (Schölkopf et al. 2001), isolation forests (Liu et al. 2008), and density estimation based methods (Shewhart 1931; Eskin 2000; Laxhammar et al. 2009). However, these methods cannot use label information. Although supervised and semi-supervised anomaly detection methods have been proposed to exploit label information (Nadeem et al. 2016; Gao et al. 2006; Das et al. 2016, 2017; Munawar et al. 2017; Pimentel et al. 2018; Akcay et al. 2018; Iwata and Yamanaka 2019), they cannot handle inexact anomaly labels.

A number of AUC maximization methods have been proposed (Cortes and Mohri 2004; Brefeld and Scheffer 2005; Ying et al. 2016; Fujino and Ueda 2016; Narasimhan and Agarwal 2017; Sakai et al. 2018) for training on class imbalanced data. However, these methods do not consider inexact labels. As a measurement related to the AUC, the partial AUC (Komori and Eguchi 2010) has been proposed. The partial AUC is the partial area under the curve with a specific false positive rate range, and it is different from the inexact AUC.

3 Preliminaries: AUC

Let $\mathcal{X}$ be an instance space, and let p_{A} and p_{N} be probability distributions over anomalous and non-anomalous instances in $\mathcal{X}$. Suppose that $a : \mathcal{X} \to \mathbb{R}$ is an anomaly score function, and anomaly detection is carried out based on its sign:

$$\mathrm{sign}(a(\mathbf{x}) - h), \tag{1}$$

where $\mathbf{x}$ is an instance and h is a threshold. The true positive rate (TPR) of anomaly score function $a(\mathbf{x})$ is the rate that it correctly classifies a random anomaly from p_{A} as anomalous,

$$\mathrm{TPR}(h) = \mathbb{E}_{\mathbf{x}^{\mathrm{A}} \sim p_{\mathrm{A}}}[I(a(\mathbf{x}^{\mathrm{A}}) > h)], \tag{2}$$

where $\mathbb{E}$ is the expectation and $I(\cdot)$ is the indicator function; $I(A) = 1$ if A is true, and $I(A) = 0$ otherwise. The false positive rate (FPR) is the rate that it misclassifies a random non-anomalous instance from p_{N} as anomalous,

$$\mathrm{FPR}(h) = \mathbb{E}_{\mathbf{x}^{\mathrm{N}} \sim p_{\mathrm{N}}}[I(a(\mathbf{x}^{\mathrm{N}}) > h)]. \tag{3}$$

The ROC curve is the plot of TPR(h) as a function of FPR(h) with different threshold h. The area under this curve (AUC) (Hanley and McNeil 1982) is computed as follows (Dodd and Pepe 2003):

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR}^{-1}(s))ds = \mathbb{E}_{\mathbf{x}^{\mathrm{A}} \sim p_{\mathrm{A}}, \mathbf{x}^{\mathrm{N}} \sim p_{\mathrm{N}}}[I(a(\mathbf{x}^{\mathrm{A}}) > a(\mathbf{x}^{\mathrm{N}}))], \tag{4}$$

where $\mathrm{FPR}^{-1}(s) = \inf\{h \in \mathbb{R} | \mathrm{FPR}(h) \leq s\}$. AUC is the rate where a randomly sampled anomalous instance has a higher anomaly score than a randomly sampled non-anomalous instance.

Given sets of anomalous instances $\mathcal{A} = \{\mathbf{x}_i^{\mathrm{A}}\}_{i=1}^{|\mathcal{A}|}$ drawn from p_{A} and non-anomalous instances $\mathcal{N} = \{\mathbf{x}_j^{\mathrm{N}}\}_{j=1}^{|\mathcal{N}|}$ drawn from p_{N}, an empirical AUC is calculated by

$$\widehat{\mathrm{AUC}} = \frac{1}{|\mathcal{A}||\mathcal{N}|} \sum_{\mathbf{x}_i^{\mathrm{A}} \in \mathcal{A}} \sum_{\mathbf{x}_j^{\mathrm{N}} \in \mathcal{N}} I(a(\mathbf{x}_i^{\mathrm{A}}) > a(\mathbf{x}_j^{\mathrm{N}})), \tag{5}$$

where $|\mathcal{A}|$ represents the size of set $\mathcal{A}$.

4 Proposed method

4.1 Inexact AUC

Let $\mathcal{B} = \{\mathbf{x}_i^{\mathrm{B}}\}_{i=1}^{|\mathcal{B}|}$ be a set of instances drawn from probability distribution p_{S}, where at least one instance is drawn from anomalous distribution p_{A}, and the other instances are drawn from non-anomalous distribution p_{N}. We define inexact true positive rate (inexact TPR) as the rate where anomaly score function $a(\mathbf{x})$ classifies at least one instance in a random instance set from p_{S} as anomalous:

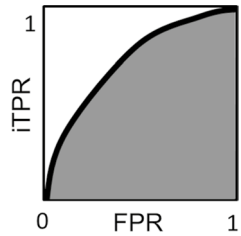


Fig. 1 Inexact AUC: area under the curve of the inexact true positive ratio as a function of the false positive rate with different threshold

$$\mathrm{iTPR}(h) = \mathbb{E}_{\mathcal{B}\sim p_{\mathrm{S}}}\left[I\left(\bigvee_{\mathbf{x}_i^{\mathrm{B}}\in\mathcal{B}}\left[a(\mathbf{x}_i^{\mathrm{B}}) > h\right]\right)\right] = \mathbb{E}_{\mathcal{B}\sim p_{\mathrm{S}}}\left[I(\max_{x_i^{\mathrm{B}}\in\mathcal{B}} a(\mathbf{x}_i^{\mathrm{B}}) > h)\right]. \tag{6}$$

We then define the inexact AUC by the area under the curve of iTPR(h) as a function of FPR(h) with different threshold h in a similar way with the AUC (4) as follows:

$$\mathrm{iAUC} = \int_0^1 \mathrm{iTPR}(\mathrm{FPR}^{-1}(s))ds = \mathbb{E}_{\mathcal{B}\sim p_{\mathrm{S}},\mathbf{x}^{\mathrm{N}}\sim p_{\mathrm{N}}}[I(\max_{\mathbf{x}_i^{\mathrm{B}}\in\mathcal{B}} a(\mathbf{x}_i^{\mathrm{B}}) > a(\mathbf{x}^{\mathrm{N}}))]. \tag{7}$$

Figure 1 shows an example of the inexact AUC. Inexact AUC is the rate where at least one instance in a randomly sampled inexact anomaly set has a higher anomaly score than a randomly sampled non-anomalous instance. When the label information is exact, i.e., every inexact anomaly set $\mathcal{B}$ contains only a single anomalous instance, the inexact AUC (7) is equivalent to the AUC (4). In addition, when anomaly score function $a(\mathbf{x})$ gives higher scores to truely anomalous instances than false anomalous instances in every inexact anomaly set, the inexact AUC is equivalent to the AUC. Therefore, the inexact AUC is a natural extension of the AUC for inexact labels.

Given a set of inexact anomaly sets $\mathcal{S} = \{\mathcal{B}_k\}_{k=1}^{|\mathcal{S}|}$, where $\mathcal{B}_k = \{\mathbf{x}_{ki}^{\mathrm{B}}\}_{i=1}^{|\mathcal{B}_k|}$, drawn from p_{S}, and a set of non-anomalous instances $\mathcal{N} = \{\mathbf{x}_j^{\mathrm{N}}\}_{j=1}^{|\mathcal{N}|}$ drawn from p_{N}, we calculate an empirical inexact AUC as follows:

$$\widehat{\mathrm{iAUC}} = \frac{1}{|\mathcal{S}||\mathcal{N}|}\sum_{\mathcal{B}_k\in\mathcal{S}}\sum_{\mathbf{x}_j^{\mathrm{N}}\in\mathcal{N}} I[\max_{\mathbf{x}_{ki}^{\mathrm{B}}\in\mathcal{B}_k} a(\mathbf{x}_{ki}^{\mathrm{B}}) > a(\mathbf{x}_j^{\mathrm{N}})]. \tag{8}$$

Figure 2 shows an example of the empirical inexact AUC. The maximum operator has been widely used for multiple instance learning methods (Andrews et al. 2003; Pinheiro and Collobert 2015; Zhu et al. 2017; Feng and Zhou 2017; Ilse et al. 2018). The proposed inexact AUC can evaluate score functions properly even with class imbalanced data by incorporating the maximum operator into the AUC framework.

4.2 Task

Suppose that we are given a set of inexact anomaly sets $\mathcal{S}$ and a set of non-anomalous instances $\mathcal{N}$ for training. Our task is to estimate anomaly scores of test instances, which are not included in the training data, so that the anomaly score is high when the test instance is anomalous, and low when it is non-anomalous. Table 1 summarizes our notation.

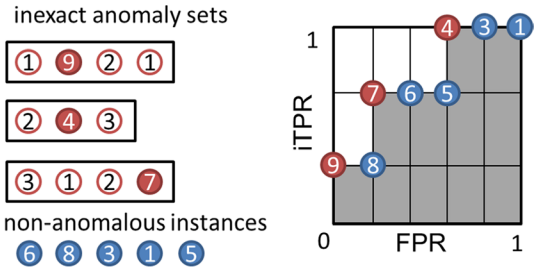


Fig. 2 Example of the empirical inexact AUC. Three sets of inexact anomaly sets and five non-anomalous instances are given. Each circle is an instance, and the value in the circle is its anomaly score. Each rectangle represents an inexact anomaly set. In each anomaly set, the instance with the maximum anomaly score is used for calculating the empirical inexact AUC

Table 1 Our notation

Symbol	Description
$\mathcal{S}$	Set of inexact anomaly sets, $\{\mathcal{B}_k\}_{k=1}^{\lvert\mathcal{S}\rvert}$
$\mathcal{B}_k$	kth inexact anomaly set, where at least one instance is anomaly, $\{\mathbf{x}_{ki}^{\mathrm{B}}\}_{i=1}^{\lvert\mathcal{B}_k\rvert}$
$\mathcal{A}$	Set of anomalous instances, $\{\mathbf{x}_i^{\mathrm{A}}\}_{i=1}^{\lvert\mathcal{A}\rvert}$
$\mathcal{N}$	Set of non-anomalous instances, $\{\mathbf{x}_j^{\mathrm{N}}\}_{j=1}^{\lvert\mathcal{N}\rvert}$
$a(\mathbf{x})$	Anomaly score of instance $\mathbf{x}$

4.3 Anomaly scores

For the anomaly score function, we use the following reconstruction error with deep autoencoders:

$$a(\mathbf{x};\boldsymbol{\theta}) = \| \mathbf{x} - g(f(\mathbf{x};\boldsymbol{\theta}_{\mathrm{f}});\boldsymbol{\theta}_{\mathrm{g}}) \|^2, \tag{9}$$

where $f(\cdot;\boldsymbol{\theta}_{\mathrm{f}})$ is an encoder modeled by a neural network with parameters $\boldsymbol{\theta}_{\mathrm{f}}$, $g(\cdot;\boldsymbol{\theta}_{\mathrm{g}})$ is a decoder modeled by a neural network with parameters $\boldsymbol{\theta}_{\mathrm{g}}$, and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{\mathrm{f}}, \boldsymbol{\theta}_{\mathrm{g}}\}$ is the parameters of the anomaly score function. The reconstruction error of an instance is likely to be low when instances similar to it often appear in the training data, and the reconstruction error is likely to be high when no similar instances are contained in the training data. With the proposed method, we can use other anomaly score functions that are differentiable with respect to parameters, such as Gaussian mixtures (Eskin 2000; An and Cho 2015; Suh et al. 2016; Xu et al. 2018), variational autoencoders (Kingma and Wellniga 2014), energy-based models (Zhai et al. 2016), and isolation forests with weight adjustment (Das et al. 2017).

4.4 Objective function

With the proposed method, parameters $\boldsymbol{\theta}$ are trained by minimizing the anomaly scores for non-anomalous instances while maximizing the empirical inexact AUC (8). To make the empirical inexact AUC differentiable with respect to the parameters, we use sigmoid

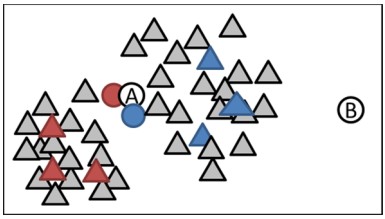


Fig. 3 Example of anomalous (circle) and non-anomalous (triangle) instances, and inexact anomaly sets (red or blue) in an instance space. Instances with identical color (red or blue) are contained in the same inexact anomaly set. White circles are test anomalous instances (Color figure online)

function $\sigma(A-B)=\frac{1}{1+\exp(-(A-B))}$ instead of step function $I(A>B)$, which is often used for a smooth approximation of the step function. Then the objective function to be minimized is given:

$$E=\frac{1}{|\mathcal{N}|}\sum_{\mathbf{x}_j^{\mathrm{N}}\in\mathcal{N}} a(\mathbf{x}_j^{\mathrm{N}})-\lambda\frac{1}{|\mathcal{S}||\mathcal{N}|}\sum_{\mathcal{B}_k\in\mathcal{S}}\sum_{\mathbf{x}_j^{\mathrm{N}}\in\mathcal{N}}\sigma\left(\max_{\mathbf{x}_{ki}^{\mathrm{B}}\in\mathcal{B}_k} a(\mathbf{x}_{ki}^{\mathrm{B}})-a(\mathbf{x}_j^{\mathrm{N}})\right), \tag{10}$$

where $\lambda\geq 0$ is a hyperparameter that can be tuned using the inexact AUC on the validation data. When there are no inexact anomaly sets or $\lambda=0$, the second term becomes zero, and the first term on the non-anomalous instances remains with the objective function, which is the same objective function with a standard autoencoder. By the unsupervised anomaly detection mechanism of the first term in (10), the proposed method can detect anomalous instances even when there are few inexact anomaly sets. When the first term in (10) is removed or $\lambda=\infty$, the proposed method corresponds to a multiple instance learning method (Dietterich et al. 1997; Maron and Lozano-Pérez 1998; Babenko et al. 2009; Wu et al. 2015; Cinbis et al. 2017), since inexact labels are used for learning classifiers, although existing multiple instance learning methods do not use AUC-based objective functions.

Figure 3 shows an example of anomalous and non-anomalous instances, and inexact anomaly sets in a two-dimensional instance space. For unsupervised methods, it is difficult to detect test anomalous instance 'A' since some instances are located around it. Unsupervised methods consider that an instance is anomalous when there are few instances around it. Since supervised methods can use label information, they can correctly detect test anomalous instance 'A'. However, supervised methods would misclassify non-anomalous instances around training instances in the inexact anomaly sets (colored triangles in Fig. 3) as anomaly since supervised methods consider all the instances in inexact anomaly sets. On the other hand, the proposed method detects 'A' as anomaly and does not misclassify non-anomalous instances by the second term in (10), where at least one instance in each inexact anomaly set has higher anomaly scores than non-anomaous instances. The reason is that, by maximizing the second term, red and blue circles are likely to have high anomaly scores because they are located closely together and the other instances in the inexact anomaly sets are surrounded by non-anomalous instances, while anomaly scores of the other instances in the inexact anomaly sets are not maximized. In addition, the proposed method detects 'B' as anomaly by incorporating an unsupervised anomaly detection mechanism of the first term in (10).

The computational complexity of calculating the objective function (10) is $O(|\mathcal{S}||\mathcal{B}| + |\mathcal{S}||\mathcal{N}|)$, where $|\mathcal{B}|$ is the average number of instances in an inexact anomaly set, the first term is for finding the maximum of anomaly scores in every inexact anomaly set, and the second term is for calculating the difference of scores between inexact anomalous instances and non-anomalous instances in the second term in (10).

5 Experiments

5.1 Data

We evaluated our proposed supervised anomaly detection method with a synthetic dataset and nine datasets used for unsupervised anomaly detection (Campos et al. 2016) [1].

The synthetic dataset was generated from a two-dimensional Gaussian mixture model shown in Figure 4a, b. The non-anomalous instances were generated from two unit-variance Gaussian distributions with mean at $(-2, 0)$ and $(2, 0)$, as shown by blue triangles in Figure 4(a). The anomalous instances were generated from a Gaussian distribution with mean $(0, -1.5)$ with a small variance and a Gaussian distribution with mean (0 ,3) with a wide variance as shown by red circles in Fig. 4a. The latter anomalous Gaussian was only used for test data, and it was not used for training and validation data as shown in Fig. 4b. We generated 500 instances from the non-anomalous Gaussians, and 200 instances from the anomalous Gaussians.

Table 2 shows the following values of the nine anomaly detection datasets: the number of anomalous instances $|\mathcal{A}|$, the number of non-anomalous instances $|\mathcal{N}|$, anomaly ratio $\frac{|\mathcal{A}|}{|\mathcal{N}|}$, and the number of attributes D. We used the nine datasets in (Campos et al. 2016) that contained enough anomalous and non-anomalous instances for generating inexact anomaly sets and evaluating methods. Each attribute was linearly normalized to range [0, 1], and duplicate instances were removed. The original datasets contained only exact anomaly labels. We constructed inexact anomaly sets by randomly sampling non-anomalous instances and an anomalous instance for each set.

We used 70% of the non-anomalous instances and ten inexact anomaly sets for training, 15% of the non-anomalous instances and five inexact anomaly sets for validation, and the remaining instances for testing. The number of instances in an inexact anomaly set was five with training and validation data, and one with test data; the test data contained only exact anomaly labels. For each inexact anomaly set, we included an anomalous instance, and the other instances were non-anomalous. For the evaluation measurement, we used AUC on test data. For each dataset, we randomly generated ten sets of training, validation and test data. We calculated the AUC for each of the ten sets, and averaged over the ten sets (Forman and Scholz 2010).

5.2 Comparing methods

We compared our proposed method with the following 11 methods: LOF, OSVM, IF, AE, KNN, SVM, RF, NN, MIL, SIF and SAE. LOF, OSVM, IF and AE are unsupervised

[1] The datasets were obtained from http://www.dbs.ifi.lmu.de/research/outlier-evaluation/DAMI/.

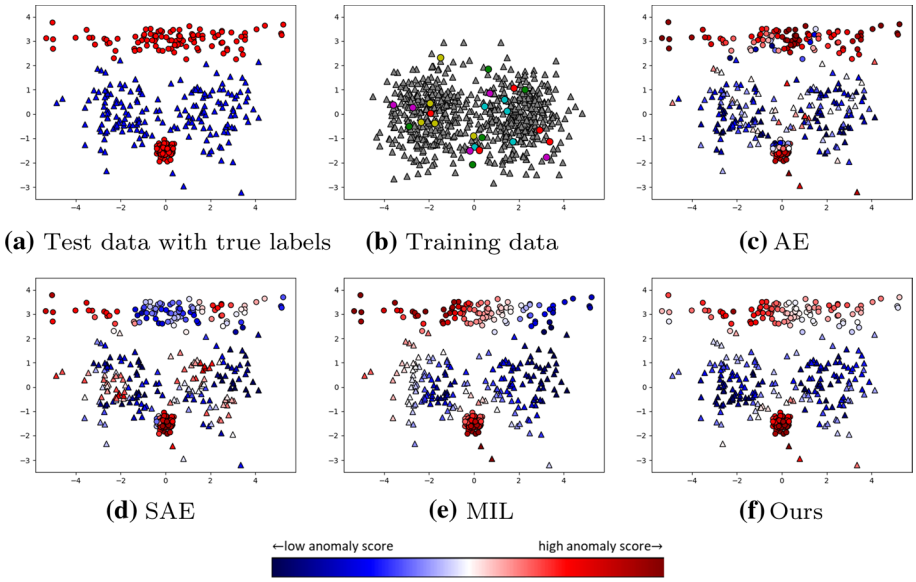


Fig. 4 Synthetic dataset and the estimated anomaly scores in the two-dimensional instance space. **a** Test data: anomalous instances are represented by red circles and non-anomalous instances are represented by blue triangles. **b** Training data: instances in the same inexact anomalous set are represented by circles with identical color, and non-anomalous instances are represented by gray triangles. In each inexact anomaly set, an instance exists in an anomalous area at the bottom center, and the other instances exist in non-anomalous areas where many non-anomalous instances exist. Note that anomalous instances at the top in the test data are not contained in the training data. **c**–**f** Estimated anomaly scores by the AE (**c**), SAE (**d**), MIL (**e**) and the proposed method (**f**). The shape indicates the true label (circle: anomalous, triangle: non-anomalous), the color indicates the estimated anomaly score; the darker red indicates the higher anomaly score, and the darker blue indicates lower anomaly score (Color figure online)

Table 2 Statistics of datasets used in our experiments. $|\mathcal{A}|$ is the number of anomalous instances, $|\mathcal{N}|$ is the number of non-anomalous instances, and D is the number of attributes

Data	$\lvert\mathcal{A}\rvert$	$\lvert\mathcal{N}\rvert$	$\frac{\lvert\mathcal{A}\rvert}{\lvert\mathcal{N}\rvert}$	D
Annthyroid	350	6666	0.053	21
Cardiotocography	413	1655	0.250	21
InternetAds	177	1598	0.111	1555
KDDCup99	246	60,593	0.004	79
PageBlocks	258	4913	0.053	10
Pima	125	500	0.250	8
SpamBase	697	2788	0.250	57
Waveform	100	3343	0.030	21
Wilt	93	4578	0.020	5

anomaly detection methods, where non-anomalous instances were used for training, but inexact anomaly sets were not used since they cannot use the label information. KNN, SVM, RF, NN, MIL, SIF, SAE and our proposed method are supervised anomaly detection methods, where both the attribute $\mathbf{x}$ and the label information are used. Since KNN, SVM, RF, NN, SIF and SAE cannot handle inexact labels, they assume that all the instances in

the inexact anomaly sets are anomalous. For hyperparameter tuning, we used the AUC scores on the validation data with LOF, OSVM, IF, AE, KNN, SVM, RF, NN, SIF and SAE, by assuming that instances in inexact anomaly sets are anomalous. Since the inexact AUC is our proposal, we did not use the inexact AUC for them. Since MIL and the proposed method used the inexact AUC for their objective functions, we used the inexact AUC scores on the validation data for hyperparameter tuning. We used the scikit-learn implementation (Pedregosa et al. 2011) with LOF, OSVM, IF, KNN, SVM, RF and NN.

LOF, which is the local outlier factor method (Breunig et al. 2000), unsupervisedly detects anomalies based on the degree of isolation from the surrounding neighborhood. The number of neighbors was tuned from $\{1, 3, 5, 15, 35\}$ using the validation data.

OSVM is the one-class support vector machine (Schölkopf et al. 2001), which is an extension of the support vector machine (SVM) for unlabeled data. OSVM finds the maximal margin hyperplane, which separates the given non-anomalous data from the origin by embedding them in a high-dimensional space by a kernel function. We used the RBF kernel, its kernel hyperparameter was tuned from $\{10^{-3}, 10^{-2}, 10^{-1}, 1\}$, and hyperparameter ν was tuned from $\{10^{-3}, 5 \times 10^{-3}, 10^{-2}, 5 \times 10^{-2}, 10^{-1}, 0.5, 1\}$.

IF is the isolation forest method (Liu et al. 2008), which is a tree-based unsupervised anomaly detection scheme. IF isolates anomalies by randomly selecting an attribute and randomly selecting a split value between the maximum and minimum values of the selected attribute. The number of base estimators was chosen from $\{1, 5, 10, 20, 30, 50, 100\}$.

AE calculates the anomaly score by the reconstruction error with the autoencoder, which is also used with the proposed method. We used the same parameter setting with the proposed method for AE, which is described in the next subsection. Although the model of the proposed method with $\lambda = 0$ is the same with that of AE, early stopping criteria were different, where the proposed method used the inexact AUC, and AE used the AUC.

KNN is the k-nearest neighbor method, which classifies instances based on the votes of neighbors. The number of neighbors was selected from $\{1, 3, 5, 15, 35\}$.

SVM is a support vector machine (Schölkopf and Smola 2002), which is a kernel-based binary classification method. We used the RBF kernel, the kernel hyperparameter was tuned from $\{10^{-3}, 10^{-2}, 10^{-1}, 1\}$, and the cost parameter was tuned from $\{10, 2, 1, 0.2, 0.1, 0.02, 0.01\}$.

RF is the random forest method (Breiman 2001), which is a meta estimator that fits a number of decision tree classifiers. The number of trees was chosen from $\{1, 5, 10, 20, 30, 50, 100\}$.

NN is a feed-forward neural network classifier. We used three layers with rectified linear unit (ReLU) activation, where the number of hidden units was selected from $\{5, 10, 50, 100\}$.

MIL is a multiple instance learning method based on an autoencoder, which is trained by maximizing the inexact AUC. We used the same parameter setting with the proposed method for the autoencoder. The proposed method with $\lambda = \infty$ corresponds to MIL, and therefore MIL is a version of the proposed method. We included MIL in comparing methods in order to demonstrate the effectiveness of the first term of our objective function (10).

SIF is a supervised anomaly detection method based on the isolation forest (Das et al. 2017), where the weights of the isolation forest are adjusted by maximizing the AUC.

SAE is a supervised anomaly detection method based on an autoencoder, where the neural networks are learned by minimizing the reconstruction error while maximizing the AUC. We used the same parameter setting with the proposed method for the autoencoder. When the proposed method uses the AUC instead of the inexact AUC in the objective function (10), the proposed method becomes the SAE.

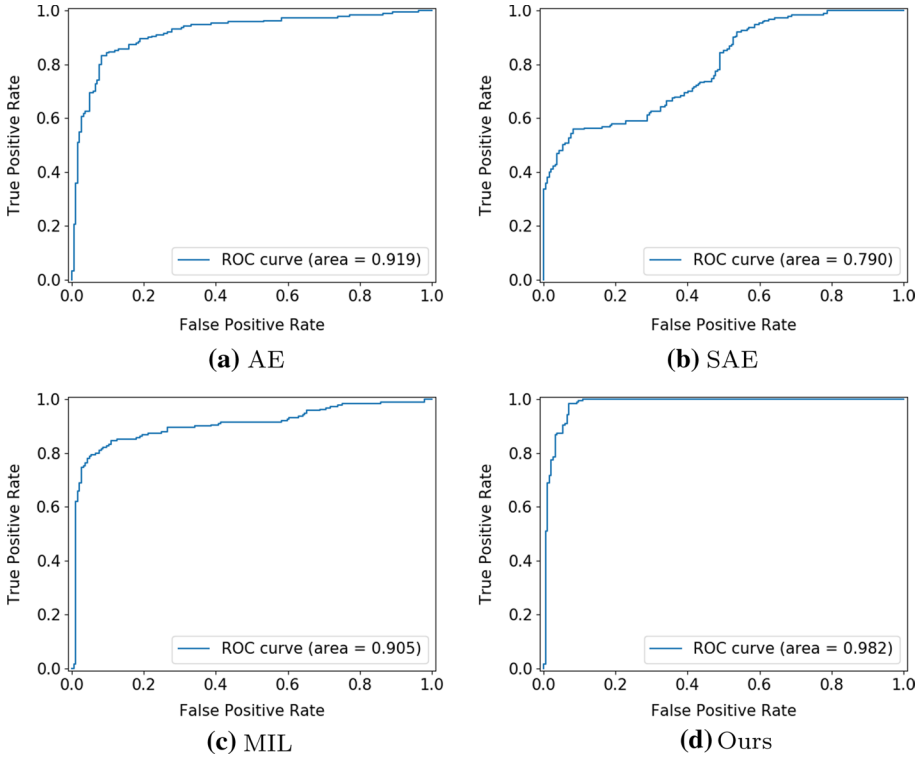


Fig. 5 ROC curve and AUC on the test synthetic dataset by **a** AE, **b** SAE, **c** MIL, and **d** the proposed method. X-axis is the false positive rate, and y-axis is the true positive rate

5.3 Settings of the proposed method

We used three-layer feed-forward neural networks for the encoder and decoder, where the hidden unit size was 128, and the output layer of the encoder and the input layer of the decoder was 16. Hyperparameter λ was selected from $\{0, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3\}$ using the inexact AUC on the validation data. The validation data were also used for early stopping, where the maximum number of training epochs was 1000. We optimized the neural network parameters using ADAM (Kingma and Ba 2015) with learning rate 10^{-3}, where we randomly sampled eight inexact anomaly sets and 128 non-anomalous instances for each batch. We implemented all the methods based on PyTorch (Paszke et al. 2017).

5.4 Results

Figure 4 shows the estimated anomaly scores by AE (c), SAE (d), MIL (e) and the proposed method (f) on the synthetic dataset. Figure 5 shows the ROC curve and test AUC by the AE (a), SAE (b), MIL (c) and the proposed method (d) on the synthetic dataset. The test AUCs were 0.919 with AE, 0.790 with SAE, 0.905 with MIL, and 0.982 with the proposed method. The AE successfully gave relatively high anomaly scores to the test anomalous instances at the top in Fig. 4c. However, since the AE is an unsupervised

method and cannot use label information, some anomalous instances at the bottom center were misclassified as non-anomaly. The SAE is a supervised method, therefore the anomalous instances at the bottom center in Fig. 4d were identified as anomaly more appropriately than the AE. However, since the SAE cannot handle inexact labels, some test non-anomalous instances, that were located around non-anomalous instances in the training inexact anomaly sets, were falsely classified as anomaly. The MIL correctly gave lower anomaly scores to test non-anomalous instances by handling inexact labels than the SAE in Fig. 4e. However, the MIL failed to correctly give high anomaly scores to unseen anomalous instances at the top. On the other hand, because the proposed method is trained by minimizing the anomaly scores for non-anomalous instances while maximizing the inexact AUC, the proposed method succeeded to detect unseen anomalous instances at the top as well as anomalous instances at the bottom center, and correctly classified test non-anomalous instances as non-anomaly in Fig. 4f.

Table 3 shows AUC on the nine anomaly detection datasets with ten inexact anomaly sets and five instances per set. Our proposed method achieved the highest AUC in most cases. Since the number of supervised labels was small, the performance of the supervised methods, KNN, SVM, RF and MIL, was not high. The proposed method outperformed them by incorporating an unsupervised method (AE) in a supervised framework. SIF and SAE also used both unsupervised and supervised anomaly detection frameworks. However, the performance was worse than the proposed method because SIF and SAE cannot handle inexact labels. Although MIL can handle inexact labels, AUC with MIL was low since it does not have an unsupervised training mechanism, i.e., it does not minimize anomaly scores for non-anomalous instances. The average computational time for training the proposed method was 1.0, 0.4, 1.4, 8.1, 0.7, 0.2, 0.5, 0.6 and 0.7 minutes with Annthyroid, Cardiotocography, InternetAds, KDDCup99, PageBlocks, Pima, SpamBase, Waveform and Wilt datasets, respectively, on computers with 2.60 GHz CPUs.

Figure 6 shows test AUCs averaged over the nine anomaly detection datasets by changing the number of training inexact anomaly sets (a), and by changing the number of instances per inexact anomaly set (b). The proposed method achieved the best performance in all cases. As the number of training inexact anomaly sets increased, the performance with supervised methods was improved. Since unsupervised methods do not use inexact anomaly sets for training, their performance did not change with the number of training inexact anomaly sets. As the number of instances per inexact anomaly set increased, AUC was decreased since the rate of non-anomalous instances in an inexact anomaly set increased. AUC with unsupervised methods also decreased since they used inexact anomaly sets in the validation data.

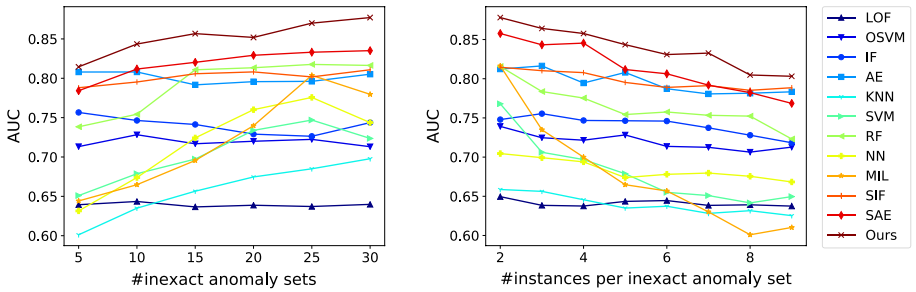
Figure 7 shows test AUC on the nine anomaly detection datasets by the proposed method with different hyperparameters λ. The best hyperparameters were different across datasets. For example, a high λ was better with the Pima dataset, a low λ was better with the PageBlocks and Wilt datasets, and an intermediate λ was better with the Annthyroid and Waveform datasets. The proposed method achieved high performance with various datasets by automatically adapting λ using the validation data to control the balance of the anomaly score minimization for non-anomalous instances and inexact AUC maximization.

We can use different neural network architectures for the autoencoder in the proposed method. Also, we can also use different anomaly score functions in the proposed method. Table 4 shows average test AUC with different autoencoder architectures and the variational autoencoder. With the variational autoencoder, we used the evidence lower bound the anomaly scores. The

Table 3 AUC on nine anomaly detection datasets with ten inexact anomaly sets and five instances per set

	Annthyroid	Cardiotocography	InternetAds	KDDCup99	PageBlocks
LOF	0.652	0.544	0.728	0.576	0.754
OSVM	0.525	**0.845**	**0.814**	0.974	0.877
IF	0.767	0.810	0.562	0.975	**0.923**
AE	0.754	0.768	**0.839**	**0.995**	**0.915**
KNN	0.556	0.677	0.574	0.823	0.702
SVM	0.744	0.692	**0.853**	0.785	0.653
RF	**0.926**	**0.866**	0.695	0.905	**0.872**
NN	0.622	0.702	0.783	0.975	0.462
MIL	0.590	0.801	**0.824**	0.714	0.609
SIF	**0.829**	**0.843**	0.622	**0.992**	**0.932**
SAE	**0.836**	0.768	**0.832**	0.924	**0.926**
Ours	**0.867**	**0.846**	**0.828**	**0.992**	**0.914**
	Pima	SpamBase	Waveform	Wilt	Average
LOF	0.601	0.546	**0.680**	0.709	0.643
OSVM	**0.686**	0.639	0.622	0.571	0.728
IF	**0.715**	0.708	**0.678**	0.611	0.750
AE	**0.678**	0.757	**0.671**	**0.895**	0.808
KNN	0.549	0.628	0.635	0.570	0.635
SVM	0.563	0.660	**0.732**	0.615	0.700
RF	0.663	**0.829**	**0.696**	**0.819**	0.808
NN	0.396	**0.782**	**0.724**	0.619	0.674
MIL	0.670	0.660	0.640	0.474	0.665
SIF	**0.706**	**0.808**	**0.723**	0.703	0.795
SAE	0.662	**0.765**	**0.728**	**0.863**	0.812
Ours	**0.713**	**0.791**	**0.746**	**0.895**	**0.844**

Values in bold typeface are not statistically better (at 5% level) from the best performing method in each dataset according to the Friedman test with the post-hoc Nemenyi test (Demšar 2006). The Average column shows the average AUC over all datasets, and the value in bold indicates that the proposed method achieved statistically better (at 5% level) AUC than the other methods across all datasets according to the Friedman test with the post-hoc Nemenyi test



(a) number of inexact anomaly sets **(b)** number of instances per inexact anomaly set

Fig. 6 AUC averaged over the nine anomaly detection datasets (**a**) with different numbers of training inexact anomaly sets and five instances per inexact anomaly set, and (**b**) with different numbers of instances per inexact anomaly set and ten training inexact anomaly sets

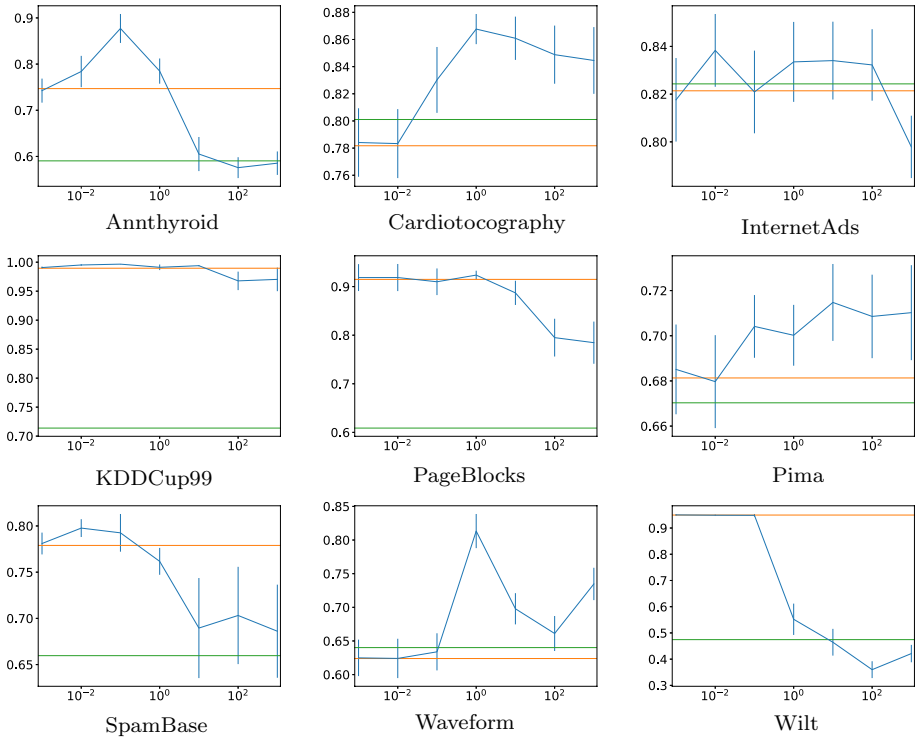


Fig. 7 AUC by the proposed method with different hyperparameters λ trained on data with ten inexact anomaly sets with five instances per set. The x-axis is hyperparameter λ and the y-axis is AUC. Errorbar shows standard error. The green and orange horizontal lines are AUC by the proposed method with $\lambda = \infty$ and $\lambda = 0$, respectively (Color figure online)

Table 4 Average AUC over all the nine anomaly detection datasets with ten inexact anomaly sets and five instances per set by different autoencoder architectures and the variational autoencoder

AE:128-16	AE:128-128-16	AE:64-16	AE:128-128-32	AE:128-128-8	VAE
0.844	0.813	0.841	0.846	0.810	0.760

AE represents the autoencoder, and VAE represents the variational autoencoder. The numbers in AEs represent the number of hidden units of the encoder network. The number of hidden units of the decoder network is the same with that of the encoder network

6 Conclusion

We proposed an extension of the AUC for inexact labels, and developed a supervised anomaly detection method for data with inexact labels. With our proposed method, we trained a neural network-based anomaly score function by maximizing the inexact AUC while minimizing the anomaly scores for non-anomalous instances. We experimentally confirmed its effectiveness using various datasets. For future work, we want to extend our framework for semi-supervised settings (Blanchard et al. 2010), where unlabeled instances, labeled anomalous and labeled non-anomalous instances are given for training. Also, we

plan to combine the inexact AUC with partial AUC (Komori and Eguchi 2010), which is the partial area under the curve with a specific false positive rate range.

References

Akcay, S., Atapour-Abarghouei, A., & Breckon, T. P. (2018). Ganomaly: Semi-supervised anomaly detection via adversarial training. In *14th Asian conference on computer vision*.

Aleskerov, E., Freisleben, B., & Rao, B. (1997). Cardwatch: A neural network based database mining system for credit card fraud detection. In *IEEE/IAFE computational intelligence for financial engineering* (pp. 220–226).

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, *2*, 1–18.

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2003). Support vector machines for multiple-instance learning. In *Advances in neural information processing systems* (pp. 577–584).

Babenko, B., Yang, M.-H., & Belongie, S. (2009). Visual tracking with online multiple instance learning. In *IEEE conference on computer vision and pattern recognition* (pp. 983–990). IEEE.

Blanchard, G., Lee, G., & Scott, C. (2010). Semi-supervised novelty detection. *Journal of Machine Learning Research*, *11*(Nov), 2973–3009.

Brefeld, U., & Scheffer, T. (2005). AUC maximizing support vector learning. In *Proceedings of the ICML workshop on roc analysis in machine learning*.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. *ACM SIGMOD Record*, *29*(2), 93–104.

Bunescu, R., & Mooney, R. (2007). Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th annual meeting of the association of computational linguistics* (pp. 576–583).

Campos, G. O., Zimek, A., Sander, J., Campello, R. J., Micenková, B., Schubert, E., et al. (2016). On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, *30*(4), 891–927.

Carbonneau, M.-A., Cheplygina, V., Granger, E., & Gagnon, G. (2018). Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, *77*, 329–353.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, *41*(3), 15.

Chen, Y., Bi, J., & Wang, J. Z. (2006). MILES: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(12), 1931–1947.

Chong, Y. S., & Tay, Y. H. (2017). Abnormal event detection in videos using spatiotemporal autoencoder. In *International symposium on neural networks* (pp. 189–196). Springer.

Cinbis, R. G., Verbeek, J., & Schmid, C. (2017). Weakly supervised object localization with multi-fold multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(1), 189–203.

Cortes, C., & Mohri, M. (2004). AUC optimization vs. error rate minimization. In *Advances in neural information processing systems* (pp. 313–320).

Das, S., Wong, W.-K., Dietterich, T., Fern, A., & Emmott, A. (2016). Incorporating expert feedback into active anomaly discovery. In *16th international conference on data mining* (pp. 853–858). IEEE.

Das, S., Wong, W.-K., Fern, A., Dietterich, T. G., & Siddiqui, M. A. (2017). Incorporating feedback into tree-based anomaly detection. In *KDD workshop on interactive data exploration and analytics*.

Davis, J., Santos Costa, V., Ray, S., & Page, D. (2007). Tightly integrating relational learning and multiple-instance regression for real-valued drug activity prediction. In *International conference on machine learning*.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*(Jan), 1–30.

Dietterich, T. G., Lathrop, R. H., & Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, *89*(1–2), 31–71.

Dodd, L. E., & Pepe, M. S. (2003). Partial AUC estimation and regression. *Biometrics*, *59*(3), 614–623.

Dokas, P., Ertoz, L., Kumar, V., Lazarevic, A., Srivastava, J., & Tan, P.-N. (2002). Data mining for network intrusion detection. In *NSF workshop on next generation data mining* (pp. 21–30).

Eskin, E. (2000). Anomaly detection over noisy data using learned probability distributions. In *International conference on machine learning*.

Feng, J., & Zhou, Z.-H. (2017). Deep miml network. In *Thirty-First AAAI conference on artificial intelligence*.

Forman, G., & Scholz, M. (2010). Apples-to-apples in cross-validation studies: Pitfalls in classifier performance measurement. *ACM SIGKDD Explorations Newsletter*, *12*(1), 49–57.

Fujimaki, R., Yairi, T., & Machida, K. (2005). An approach to spacecraft anomaly detection problem using kernel feature space. In *International conference on knowledge discovery in data mining* (pp. 401–410).

Fujino, A., & Ueda, N. (2016). A semi-supervised AUC optimization method with generative models. In *16th international conference on data mining* (pp. 883–888). IEEE.

Gao, J., Cheng, H., & Tan, P.-N. (2006). A novel framework for incorporating labeled examples into anomaly detection. In *Proceedings of the 2006 SIAM international conference on data mining* (pp. 594–598). SIAM.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, *143*(1), 29–36.

Herrera, F., Ventura, S., Bello, R., Cornelis, C., Zafra, A., Sánchez-Tarragó, D., et al. (2016). *Multiple Instance Learning: Foundations and Algorithms*. Berlin: Springer.

Hodge, V., & Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Ntelligence Review*, *22*(2), 85–126.

Idé, T., & Kashima, H. (2004). Eigenspace-based anomaly detection in computer systems. In *International conference on knowledge discovery and data mining* (pp. 440–449).

Ilse, M., Tomczak, J., & Welling, M. (2018). Attention-based deep multiple instance learning. In *International conference on machine learning* (pp. 2132–2141).

Iwata, T., & Yamanaka, Y. (2019). Supervised anomaly detection based on deep autoregressive density estimators. arXiv preprint arXiv:1904.06034

Kingma, D. P., & Ba, J. (2015). ADAM: A method for stochastic optimization. In *International conference on learning representations*.

Kingma, D. P., & Wellniga, M. (2014). Auto-encoding variational Bayes. In *2nd international conference on learning representations*.

Komori, O., & Eguchi, S. (2010). A boosting method for maximizing the partial area under the ROC curve. *BMC Bioinformatics*, *11*(1), 314.

Laxhammar, R., Falkman, G., & Sviestins, E. (2009). Anomaly detection in sea traffic—A comparison of the Gaussian mixture model and the kernel density estimator. In *International conference on information fusion* (pp. 756–763).

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *Proceeding of the 8th IEEE international conference on data mining* (pp. 413–422). IEEE.

Markou, M., & Singh, S. (2003). Novelty detection: A review. *Signal Processing*, *83*(12), 2481–2497.

Maron, O., & Lozano-Pérez, T. (1998). A framework for multiple-instance learning. In *Advances in neural information processing systems* (pp. 570–576).

Mukkamala, S., Sung, A., & Ribeiro, B. (2005). Model selection for kernel based intrusion detection systems. In *Adaptive and natural computing algorithms* (pp. 458–461). Springer.

Munawar, A., Vinayavekhin, P., & De Magistris, G. (2017). Limiting the reconstruction capability of generative neural network using negative learning. In *27th international workshop on machine learning for signal processing*. IEEE.

Nadeem, M., Marshall, O., Singh, S., Fang, X., & Yuan, X. (2016). Semi-supervised deep neural network for network intrusion detection. In *KSU conference on cybersecurity education, research and practice*.

Narasimhan, H., & Agarwal, S. (2017). Support vector algorithms for optimizing the partial area under the ROC curve. *Neural Computation*, *29*(7), 1919–1963.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in PyTorch. In *NIPS autodiff workshop*.

Patcha, A., & Park, J.-M. (2007). An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, *51*(12), 3448–3470.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pimentel, T., Monteiro, M., Viana, J., Veloso, A., & Ziviani, N. (2018). A generalized active learning approach for unsupervised anomaly detection. arXiv preprint arXiv:1805.09411.

Pinheiro, P. O., & Collobert, R. (2015). From image-level to pixel-level labeling with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1713–1721).

Rapaka, A., Novokhodko, A., & Wunsch, D. (2003). Intrusion detection using radial basis function network on sequences of system calls. *International Joint Conference on Neural Networks*, *3*, 1820–1825.

Sabokrou, M., Fathy, M., & Hoseini, M. (2016). Video anomaly detection and localisation based on the sparsity and reconstruction error of auto-encoder. *Electronics Letters*, *52*(13), 1122–1124.

Sakai, T., Niu, G., & Sugiyama, M. (2018). Semi-supervised AUC optimization based on positive-unlabeled learning. *Machine Learning*, *107*(4), 767–794.

Sakurada, M., & Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2nd workshop on machine learning for sensory data analysis*. ACM.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, *13*(7), 1443–1471.

Schölkopf, B., Smola, A. J., et al. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge: MIT press.

Shewhart, W. A. (1931). *Economic Control of Quality of Manufactured Product*. Milwaukee: ASQ Quality Press.

Singh, S., & Silakari, S. (2009). An ensemble approach for feature selection of cyber attack dataset. arXiv preprint arXiv:0912.1014

Suh, S., Chae, D. H., Kang, H.-G., & Choi, S. (2016). Echo-state conditional variational autoencoder for anomaly detection. In *International joint conference on neural networks*(pp. 1015–1022).

Wong, W.-K., Moore, A. W., Cooper, G. F., & Wagner, M. M. (2003). Bayesian network anomaly pattern detection for disease outbreaks. In *International conference on machine learning* (pp. 808–815).

Wu, J., Yu, Y., Huang, C., & Yu, K. (2015). Deep multiple instance learning for image classification and auto-annotation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3460–3469).

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., et al. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *World wide web conference* (pp. 187–196).

Yamanishi, K., Takeuchi, J.-I., Williams, G., & Milne, P. (2004). On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, *8*(3), 275–300.

Ying, Y., Wen, L., & Lyu, S. (2016). Stochastic online AUC maximization. In *Advances in neural information processing systems* (pp. 451–459).

Zhai, S., Cheng, Y., Lu, W., & Zhang, Z. (2016). Deep structured energy based models for anomaly detection. In *International conference on machine learning* (pp. 1100–1109).

Zhang, Q., Goldman, S. A., Yu, W., & Fritts, J. E. (2002). Content-based image retrieval using multiple-instance learning. In *International conference on machine learning*.

Zhou, C., & Paffenroth, R. C. (2017). Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 665–674). ACM.

Zhou, Z.-H., Sun, Y.-Y., & Li, Y.-F. (2009). Multi-instance learning by treating instances as non-iid samples. In *Proceedings of the 26th annual international conference on machine learning* (pp. 1249–1256).

Zhu, W., Lou, Q., Vang, Y. S., & Xie, X. (2017). Deep multi-instance networks with sparse label assignment for whole mammogram classification. In *International conference on medical image computing and computer-assisted intervention* (pp. 603–611).