Anomaly Detection Using Replicator Neural Networks Trained on Examples of One Class

Conference Paper · December 2014

DOI: 10.1007/978-3-319-13563-2_27

3 authors, including:

Anh Dau
University of California, Riverside
17 PUBLICATIONS **208** CITATIONS

SEE PROFILE

Andy Song
RMIT University
96 PUBLICATIONS **679** CITATIONS

SEE PROFILE

Anomaly Detection Using Replicator Neural Networks Trained on Examples of One Class

Hoang Anh Dau, Vic Ciesielski, and Andy Song

RMIT University, Melbourne 3000, Victoria, Australia
http://www.rmit.com.au

Abstract. Anomaly detection aims to find patterns in data that are significantly different from what is defined as normal. One of the challenges of anomaly detection is the lack of labelled examples, especially for the anomalous classes. We describe a neural network based approach to detect anomalous instances using only examples of the normal class in training. In this work we train the net to build a model of the normal examples, which is then used to predict the class of previously unseen instances based on reconstruction error rate. The input to this network is also the desired output. We have tested the method on six benchmark data sets commonly used in the anomaly detection community. The results demonstrate that the proposed method is promising for anomaly detection. We achieve F-score of more than 90% on 3 data sets and outperform the original work of Hawkins et al. on the Wisconsin breast cancer set.

Keywords: artificial neural networks, replicator neural network, autoencoder, anomaly detection, one-class learning.

1 Introduction

The term anomaly or outlier has come from the field of statistics, wherein anomaly detection has been studied since at least the 19th century. Anomalies are also referred to as an outlier in the literature. The most quoted definition of anomaly comes from Hawkins' 1980 book: "An outlier is an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism" [1]. Many real world problems can be formulated as an anomaly detection task. For example, one way to detect possible network intrusion is to look for abnormal patterns in network traffic. Detecting anomalies is important because anomalous data often implies negative or even destructive consequences. Alternatively it may represent a positive thing such as rich mineral deposit in magnetic field data. Detecting and then removing anomalies can improve the performance of classification, clustering and regression algorithms, because even a single anomalous value can significant bias these algorithms. For example, Chen recently showed that a single anomalously smooth exemplar will condemn semi-supervised time series classification algorithms to fail [2].

G. Dick et al. (Eds.): SEAL 2014, LNCS 8886, pp. 311–322, 2014.

Different names for the anomaly detection problem as used in literature are: outlier detection, surprise detection, discord detection, deviation detection, exception detection and one-class classification. The machine learning version of anomaly detection aims to build a model that can differentiate anomalous instances from the rest of the data. Techniques can operate in one class or multi-class setting. In one class setting, all anomalous instances are treated as one class. In contrast, in multi-class setting, the goal is to learn different classes of an anomalous nature. Various methods have been proposed for anomaly detection, using concepts drawn from multiple disciplines.

Anomaly detection remains a challenging research topic. It is often difficult to specify a concrete definition of normal regions as opposed to abnormal regions. For that reason, techniques in anomaly detection are usually domain-specific. The lack of labelled data is a common problem. This gives rise to the popularity of techniques operating in unsupervised or semi-supervised mode. Unsupervised anomaly detection does not require training data. However, it makes implicit assumption about the unbalanced distribution of the data set. Thus, the techniques report high false alarm rate if this condition does not hold. On the other hand, techniques that operate in semi-supervised mode require only normal examples for training. In practice, it is often more feasible to collect instances of the normal class than to obtain a comprehensive set of anomalies. This is the approach that our method uses.

Our method is inspired by studies of Hawkins et al., who first proposed Replicator Neural Network (RNN) for anomaly detection [3], [4]. The idea is to train a network capable of replicating the input as its output through a data compression and decompression process. The technique can detect most of the outliers in large multi-variate data sets and can work directly on raw data. However, the network that Hawkins et al. employed consists of 3 hidden layers and this increases the number of parameters to tune considerably. Furthermore, the method only produces an anomaly score, which gives an estimation of the anomalous degree of the pattern. It, however, does not enable direct decision-making without specific domain knowledge. In our study, we show that a network with only one single hidden layer can perform just as well or even better. The outputs of our method offers both an anomaly score and a corresponding threshold value for label assignment.

Our method considers the detection of a point anomaly in a single class setting. The goal is to decide whether an individual instance is anomalous with respect to the rest of the data. To achieve that objective, we train the neural networks to reconstruct instances of the normal class. In testing, we expose the trained network to examples of both classes. We expect the network to replicate the normal instances with marginal error while handling anomalous instances poorly, indicating by the high error rate. Therefore the error value can guide the judgement on the anomalous degree of each observation. We then apply cut-off on sorted error values to assign an anomaly label. We tested the technique on a collection of 6 data sets published in anomaly detection literature. The result shows that our method is competitive or better than other published results.

1.1 Research Questions

1. How can replicator neural network with only one hidden layer be used to detect anomalies?
2. How well does this technique work on different data sets?

Our original intention was to apply the replicator neural network technique to data sets used in previously published works on anomaly detection and compare the performance with that of the previously published methods. This proved to be surprisingly difficult. Many data sets are not available or poorly described. Measures of performance were ad hoc and inconsistent with authors choosing metrics that favour their technique.

2 Related Work

2.1 Anomaly Detection

The broader spectrum of anomaly detection techniques employs concepts from various disciplines Each technique has relative strengths and weaknesses. Anomaly detection techniques can be categorised in different ways. In terms of outputs, some techniques produce an anomaly score while the others produce an anomaly label. In terms of data type, broadly speaking, there are two groups, one works on categorical data whilst the other works on continuous data. There exists techniques to convert data between symbolic and continuous, which enable anomaly detection methods which are not originally designed for one kind of data to work on the other. Almost all anomaly detection techniques require some kinds of distance measures. An instance is assessed based on its relative distance from other instances. A comprehensive survey of research on anomaly detection is presented in [5].

2.2 One-Class Learning

Moya et al provided one of the first papers on one-class learning [6]. One-class classification is similar to binary classification, in which the goal is to categorize the instances into two distinct groups. The difference is that in one-class learning, the training set contains the objects of one class only. Training set of traditional classification problem, on the other hand, must include objects from all the classes. The machine learning version of anomaly detection problem is fundamentally one-class learning. The detector is trained to learn the normal examples and everything else can be classified as outliers.

2.3 Neural Network for Anomaly Detection

Artificial Neural networks (ANN) have been used for classification-based anomaly detection [5]. One of the main advantages is that the technique makes no assumption about the data distribution. Moreover, it is capable of inducing a predictive non-linear model and handles well data sets of high dimension. ANN has been

used for for both one-class and multi-class anomaly detection. A basic anomaly detection technique using ANN in one-class setting operates in two steps. A neural network is first trained on normal examples to learn the different normal classes. Each test instance is then provided as input to this trained network. The network is supposed to reject an input if it is an outlier.

2.4 Replicator Neural Network for Anomaly Detection

The term replicator neural networks (RNN) is intended for all auto-associative neural networks with compressed internal representations. RNN is also known in literature as autoencoder. The basic idea of RNN is that the input vector is also the target vector; we simply duplicate the input to be the output. The output layer therefore has equally many nodes as the input layer. Instead of training to predict a target value y given an input x, the network is trained to reconstruct its own input x. The weights of the RNN is driven by the goal to minimize the reconstruction error rate. Fig.1 displays a visualisation of a RNN network. One of the early studies of RNN is done by Hecht-Nielsen almost two decades ago [7], in which the author discusses two theorems regarding the data compression capability of the 3-hidden-layer RNN. Hecht-Nielsen himself credited the first serious study of RNN to Kohonen with the work on Self-Organizing Map.

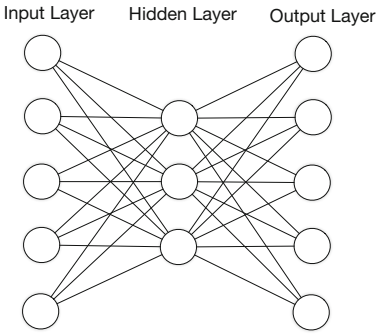

Fig. 1. A replicator neural network with one hidden layer consisting of 3 units. The input and output layer have 5 units each.

RNN was first proposed for outlier detection by Hawskin et al [3]. Their network is a feed-forward multi-layer perceptron with three hidden layers sandwiched between an input layer and an output layer. They use the *tanh* activation function for the two outer hidden layers and a staircase-like activation function for the middle hidden layer, which fit the data points into a number of clusters. They train the net to reconstruct the normal instances and reason that the trained net should replicate the unseen normal instances relatively well. Therefore if an input pattern is poorly reconstructed, indicating by the high

reconstruction error, it is likely to be an anomaly. Their method produces an anomaly score for each record, which is the reconstruction error value. They test the technique on two public data sets, namely the Wisconsin breast cancer set and the 1999 KDD Cup network intrusion detection set.

Toth et al. used RNN for outlier modelling in speech recognition [8]. Without RNN, they face the lack of labelled anomaly for training and therefore have to generate synthetic outlier examples. RNN eliminates this task and still offers comparable performance. They point out that using three hidden layers is unnecessary and that the 3 and 4-layer version RNN produce similar result. They find the traditional sigmoid activation function converge the best and use it for all layers.

In a study on texture retrieval but can be easily formulated as an anomaly detection problem, Ciesielski et al. [9] use RNN to find regions of a texture of interest in arbitrary images. This study also finds that using only 1 hidden layer gives smallest train error and test error. The number of units in the hidden layer gave varied accuracies; generally choosing approximately the same number of hidden nodes as the number of inputs gave a good trade off between accuracy and training time.

3 Learning an Anomaly Detector

3.1 Problem Definition

We formulate anomaly detection with RNN as a one-class learning problem. The network is trained to learn what is normal in order to detect the abnormal. The trained network should have little problem reconstructing normal examples, showing by the low reconstruction error. On the other hand, it should encounter difficulty in reconstructing anomalous instances, indicating by the high reconstruction error. The reconstruction error therefore can be used as the anomaly score for each instance. The maximum error at the end of the training process can be used as a threshold value for outputting binary predicate. All the instances with corresponding outlier score below this cut-off value belong to the normal class and the rest is anomalous.

3.2 Data Preparation

We use raw data directly without any feature extraction. For some public data sets, the data may have gone through some kinds of pre-processing. All the attributes are normalized to the range $[0, 1]$ using the Weka software tool-kit [10]. A pattern to be given to the network is created by representing each instance as a feature vector and duplicate the input as output. Instances with missing values and data label entries are removed.

For each data set, the normal instances and anomalous instances are separated to form a normal set and a anomalous set. A training set is created by randomly sampling a number of instances from the normal set. A validation set is created

by randomly sampling a number of instances from the normal set (excluding the ones used in training). A test set is created by randomly sampling a number of instances from the normal set (excluding the one used for training and validation) and a number of instances from the anomalous set. Ground-truth is only used after the testing phase to evaluate the performance.

3.3 The Training Algorithm

Generating an anomaly detector involves training a neural network and then finding a suitable threshold. The methodology is:

1. Generate a training set of N normal examples.
2. Generate a validation set of M normal examples.
3. Create a three-layer feed-forward network with random initial weights. The number of units in the input-output layer is equal to the number of variables in the data set. The number of hidden neurons is determined empirically.
4. Use back-propagation to train the network. Training ceases when the error on the validation set begins to rise.
5. Choose the maximum error in training to be the threshold.

4 Experiments and Results

We use the JavaNNS system (Java Neural Network Simulator) [11] for all the experiments. We leave most of the default parameter setting untouched.

4.1 Evaluation Metrics

For each data set, we report a range of evaluation metrics including the F-score, accuracy, true positive rate, true negative rate and top-p accuracy. The choice of F-score is to take into account the possible unbalanced distribution of some data sets. Top-p accuracy is a metric used in [5] to evaluate 10 state-of-the-art anomaly detection techniques. Take p to be the number of true anomalies in the entire test set and t to be the number of true anomalies in the top p records based on anomaly score, top-p accuracy is equal to $\frac{t}{p}$.

4.2 Data Sets

One of our research goals is to see how the technique performs on a wide range of real data sets, and provide a relative comparison with other techniques. For this reason, we selected data sets that have been used in published anomaly detection literature. Table 2 lists the statistic summary for each data set and Table 3 displays the test result.

Table 1. Summary of each data set including name, number of attributes, attribute type, total number of instances, number of instances in training set, validation set and test set respectively. All data sets are of numeric value. These data sets were used in published anomaly detection works.

Name	Patterns	Attributes	Anomalies	Training	Validation	Test
Breast Cancer	683	9	239	150	50	483
Ionosphere	351	34	126	100	30	221
Musk	7074	166	1224	3000	1000	3074
Biomed	194	4	67	50	50	94
Shape1	30	162	10	7	3	20
Shape2	70	162	10	20	10	40

Winscosin Breast Cancer Data Set [12]. The data set consists of 699 instances, of which 458 are benign (65.5%) and 241 malignant (34.5%). The data has 9 real-valued features. In the context of this paper, we consider benign to be of normal class and malignant to be of the anomalous class.

Ionosphere Data Set [13]. This is radar data collected by a system in Goose Bay, Labrador. "Good" radar returns shows some types of structure in the ionosphere while "bad" radar returns do not. We label 251 good radar as normal instances and 126 bad radar as anomalies. All 34 attributes are numeric.

Musk Data Set [14]. The data set consists of 7074 instances. Each is a 166-dimensional feature vector representing a conformation of a molecule. We label the musk instances as anomalies (1255) and non-musk ones as normal (5857).

Biomed Data Set [15]. This data set consists of 194 examples with 4 attributes corresponding to measurements made on blood samples. Normal examples are observations from healthy patients. Abnormal examples are from patients with a rare genetic disease.

Shape Anomaly Data Set [16]. The full data set is originally from the UCR time series database [17]. We only consider two subsets used in [5] and available at [16]. Each instance represents a shape converted into one dimensional time series. The normal time series correspond to one or more shapes, while the anomalous time series correspond to other shapes. **Shape2** is considered more complex than **Shape1**. Each shape time series is 1614 data-point long. We decide to down sample the features as their number is too large (more than 1000). Instead of using all 1614 features, we choose every 10th data points, resulting in 161 attributes left.

Table 2. Result for all data sets: F-score, Accuracy, True Positive Rate, True Negative Rate, the threshold value based on maximum error, Top-p Accuracy and the network architecture used. The numbers in bold indicate high performance.

Name	F-score	Accuracy	TPR	TNR	Thresh.	Top-p A.	Network
Cancer	**94.29**%	94.36%	93.30%	95.42%	0.0634	94.53%	9-9-9
Ionosphere	**92.00**%	90.74%	91.27%	90.00%	0.2138	91.27%	34-34-34
Musk	41.93%	67.76%	28.76%	94.28%	0.4453	54.49%	166-166-166
Biomed	55.67%	54.25%	40.30%	88.89%	0.0238	74.63%	4-7-4
Shape1	**95.24**%	95.00%	100.00%	90.00%	1.1901	100.00%	162-162-162
Shape2	51.28%	52.50%	100.00%	36.67%	2.3580	70.00%	162-162-162

4.3 Choosing the Number of Units for the Single Hidden Layer

The selection of the number of hidden neurons has a couple of implications. If it is too large, the system will be over specified. Conversely, if it is too small, the system can become over-generalized and therefore poorly infers specific cases.

We find that the choice of hidden unit quantity significantly affect the technique accuracy. Table 4 shows varied measurement metrics for different network architectures on the breast cancer data set. For each scenario, we run the experiments 5 times and notice the result stays consistent. Generally, having approximately the same number of hidden units as the number of input-output units give high performance (F-score of more than 90%). Interestingly, further experiments on different data sets show that having the number of hidden units equal to the input-output units consistently yields good detection rate even though depending on the data set, it may not be the optimum architecture.

Table 3. Test result for the breast cancer data set with different network architectures, training uses standard back-propagation and stops at 2000 epochs. The numbers in bold indicate high performance.

Network	F-score	Accuracy	TPR	TNR	Top-p A.
9-0-9 net	78.16%	81.21%	67.37%	95.00%	76.00%
9-1-9 net	61.85%	72.44%	44.77%	100.00%	96.00%
9-3-9 net	89.00%	90.00%	81.17%	98.75%	95.80%
9-9-9 net	**94.29**%	94.36%	93.30%	95.42%	94.34%
9-12-9 net	**93.53**%	93.53%	93.73%	93.33%	93.70%

4.4 Comparison with Other Works

We find it hard to systematically compare different techniques because the community has not converged on a single quality metric. For comparison purpose, we compute the same metrics as the original authors when possible.

When the comparison is impossible, we refer to the baseline accuracy used in [18]. Baseline accuracy for a given data set is the ratio $\frac{n}{(m+n)}$, in which n is the number of anomalies and m is the number of normal instances in the test set respectively. For the method to be considered effective, its top-p accuracy must be significantly better than its corresponding baseline accuracy.

Breast Cancer Data Set Used by Hawkins et al. [3]. We recreate their version of the breast cancer data by choosing one in every six malignant records to form an unbalanced distribution. The resultant data set has 39 malignant (8%) and 444 benign (92%). The authors, however, do not mention the specific number of instances used for training, validation and test respectively. For our experiment, we use 150 normal patterns for training, 50 normal patterns for validation. The test set includes 244 normal patterns and all 39 anomalous patterns. Overall, our method is better at detecting anomalies for this data set. We report the best result in Table 6. Our top 16 records are all anomalies while Hawkins' only contains 11 true anomalies. Our technique captures all the malignant in the top 48 records while Hawkins' method only detects 89.74%. They instead need to examine the top 112 to cover all the malignant cases present.

The Musk and Ionosphere Data Set Used by Aggarwal et al. [19]. The authors' original purpose is to compare performance of brute-forte search and evolutionary search in detecting anomaly. Only the search time for each algorithm is reported and no other detection rate was given. We therefore turn to baseline accuracy described in the Evaluation Metrics section to assess the technique's effectiveness. We achieved 54.49% top-p accuracy for the Musk set given the baseline accuracy of 39.81%. Our technique captures most of the anomalies in the top ranked records of the Ionosphere set, gaining 91.27% top-p accuracy given the baseline accuracy of 57.01%.

The Biomed Data Set Used by Bennett et al. [20]. The authors report performance as a variable dependent on the σ value. The best case is when $\sigma = 1.1$, for which the technique correctly labels 2 out of 27 normal instances and 57 out of 67 anomalous instances. According to our calculation, this gives a F-score of 76.50%. Our technique scores 55.67%. However, it does not necessarily means their technique is better than ours. Reporting performance for different σ value is no different from reporting performance for different threshold values. We can simply adjust the threshold to have a better detection rate. Note that our threshold is a product of the training process, not a value tuned in testing.

Shape1 and Shape2 Data Set Used by Varun [18]. These are among 19 data sets used to evaluate 10 existing state of the art anomaly detection techniques for time series data. Shape1 is relatively easy and most techniques give 100% top-p accuracy. The worst result is around 90%. Our method also obtains 100% top-p accuracy for this set. Shape2 is harder and most techniques

Table 4. A comparison of Hawkins et al. result and our method's result on the breast cancer data set. The first and the second column show the number of malignant present in the top ranked records. The third column displays the corresponding percentage of malignant out of all malignant there are in the whole data set. Our network is trained by standard backpropagation through 500 iterations, 9-9-9 network architecture.

Hawkins' method result			Our method's result		
Malignant	Record	% Malignant	Malignant	Record	% Malignant
0	0	0.00%	0	0	0.00%
3	4	7.69%	4	4	10.25%
6	8	15.38%	8	8	20.51%
11	16	28.21%	16	16	41.02%
18	24	46.15%	23	24	58.97%
25	32	64.10%	29	32	74.36%
30	40	76.92%	36	40	92.30%
35	**48**	**89.74%**	**39**	**48**	**100.00%**
36	56	92.31%	39	56	100.00%
36	64	92.31%	39	64	100.00%
38	72	97.44%	39	72	100.00%
38	80	97.44%	39	80	100.00%
38	100	97.44%	39	100	100.00%
39	112	100.00%	39	112	100.00%

struggle. The best performance is around 90% and the worst is only around 20%. Our method stays competitive with 70% top-p accuracy. Note that our method is not specifically designed for detecting anomaly in a time series database. Time series data has the temporal correlation between data points which does not exist in normal record data.

4.5 Discussion

We notice the lack of benchmark data for anomaly detection. Each paper uses distinct data sets and evaluation metrics, making a fair assessment challenging. One recent paper describes an anomaly detection benchmark from real data [21]. With 19 data sets from the UCI repository, the authors generate 4,369 problem set replicates of various difficulty levels. Unfortunately, these problem sets are not publicly available.

The experiment result demonstrates that using RNN with only one hidden layer is a promising approach for anomaly detection. Even though the optimum number of hidden neurons is dependent on the data dimensionality, we have managed to narrow down to an optimum range. We suggest having this number slightly fewer than, equal to or slightly higher than the number of input-output units are all reasonably good options. When an exhaustive search is impossible, we recommend using the same number of units for all three RNN layers. This finding is somewhat surprising because the intuition is that the number of units

in the hidden layer should be smaller than that of the two outer layers to enable data compression and helps the network generalises unseen examples.

In terms of training iteration, we note that most of the time the network converges at 2000 epochs or less. The maximum reconstruction error in training varies for each data set. Therefore, there is no universal threshold value for all. At the same time, we see no correlation between the performance and the corresponding threshold value. We speculate that the threshold intensity may just reflect the diversity of the training examples. We find that threshold alone cannot effectively distinguish all the anomalies because some atypical instances consistently has low reconstruction error.

5 Conclusions

We have shown how replicator neural networks can be used for anomaly detection. This can be done with a three layer feed-forward network with one single hidden layer. The input to this network is also the desired output. The number of units in the input and output layer corresponds to the number of the data attributes. Only normal instances are used for training. The output of the training process is a predictive model and a corresponding threshold value. Our method not only gives a ranked estimation of the anomalous degree of each instance but also provides anomaly label for direct decision-making.

While our original intention was to examine the performance of the method on a wide range of previously used data sets, this proved to be very difficult. We, however, were able to perform comparisons on 6 data sets published in the anomaly detection literature. The technique achieves high F-score of above 90% on 3 out of 6 data sets. Comparison with other works shows that our method is competitive and sometimes better. Our method outperforms the Hawkins et al. original work on using RNN for anomaly detection with the Wisconsin breast cancer set.

Future work would be to investigate other alternatives of choosing a threshold value. For the data sets that the technique does not work quite well, we plan to investigate whether increasing the number of hidden layers can help. We also would like to have the method tested on more data sets of different dimensions and application domains.

References

1. Hawkins, D.M.: Identification of outliers, vol. 11. Springer (1980)
2. Chen, Y., Hu, B., Keogh, E., Batista, G.E.: Dtw-d: time series semi-supervised learning from a single example. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 383–391. ACM (2013)
3. Hawkins, S., He, H., Williams, G.J., Baxter, R.A.: Outlier detection using replicator neural networks. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) DaWaK 2002. LNCS, vol. 2454, pp. 170–180. Springer, Heidelberg (2002)

4. Williams, G., Hawkins, S., Gu, L., Baxter, R., He, H.: A comparative study of rnn for outlier detection in data mining. In: IEEE International Conference on Data Mining, pp. 709–709. IEEE Computer Society (2002)
5. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Computing Surveys (CSUR) 41(3), 15 (2009)
6. Moya, M.M., Hush, D.R.: Network constraints and multi-objective optimization for one-class classification. Neural Networks 9(3), 463–474 (1996)
7. Hecht-Nielsen, R.: Replicator neural networks for universal optimal source coding. Science-New York Then Washington, 1860–1860 (1995)
8. Tóth, L., Gosztolya, G.: Replicator neural networks for outlier modeling in segmental speech recognition. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3173, pp. 996–1001. Springer, Heidelberg (2004)
9. Ciesielski, V., Ha, V.P.: Texture detection using neural networks trained on examples of one class. In: Nicholson, A., Li, X. (eds.) AI 2009. LNCS, vol. 5866, pp. 140–149. Springer, Heidelberg (2009)
10. Weka, `http://www.cs.waikato.ac.nz/ml/weka/downloading.html/` (online; accessed July 23, 2014)
11. Javanns: Java neural network simulator, `http://www.ra.cs.uni-tuebingen.de/software/JavaNNS/` (online; accessed July 12, 2014)
12. Breast cancer wisconsin (prognostic) data set, `http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/` (online; accessed July 9, 2014)
13. Ionosphere data set, `https://archive.ics.uci.edu/ml/datasets/Ionosphere/` (online; accessed July 17, 2014)
14. Musk (version 2) data set, `https://archive.ics.uci.edu/ml/datasets/Musk+Version+2/` (online; accessed July 17, 2014)
15. Musk (version 2)data set, `http://lib.stat.cmu.edu/datasets/` (online; accessed July 18, 2014)
16. Chandola time series data sets, `http://www-users.cs.umn.edu/~chandola/timeseries/` (online; accessed July 9, 2014)
17. The ucr time series classification/clustering page, `http://www.cs.ucr.edu/~eamonn/time_series_data/` (online; accessed July 9, 2014)
18. Chandola, V.: Anomaly detection for symbolic sequences and time series data. Ph.D. dissertation, University of Minnesota (2009)
19. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. ACM Sigmod Record 30(2), 37–46 (2001)
20. Campbell, C., Bennett, K.P.: A linear programming approach to novelty detection. In: Proceedings of the 2000 Conference on Advances in Neural Information Processing Systems 13, vol. 13, p. 395. MIT Press (2001)
21. Emmott, A.F., Das, S., Dietterich, T., Fern, A., Wong, W.-K.: Systematic construction of anomaly detection benchmarks from real data. In: Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description, pp. 16–21. ACM (2013)