Anomaly Detection on Time Series

Mingyan Teng
Department of Mathematics, Bohai University, Jinzhou, Liaoning, China 121000
Email: tengmingyan@gmail.com

***Abstract*—The problem of anomaly detection on time series is to predict whether a newly observed time series novel or normal, to a set of training time series. It is very useful in many monitoring applications such as video surveillance and signal recognition. Based on some existing outlier detection algorithms, we propose an instance-based anomaly detection algorithm. We also propose a local instance summarization approach to reduce the number of distance computation of time series, so that abnormal time series can be efficiently detected. Experiments show that the proposed algorithm achieves much better accuracy than the basic outlier detection algorithms. It is also very efficient for anomaly detection of time series.**

***Keywords*—Anomaly detection; time series.**

I. Introduction

Time series can be used to represent shapes of objects or trajectories of moving objects [2], which can be further applied for identifying objects and phenomena in applications such as video surveillance and signal recognition. Anomaly detection in these applications aims to automatically detect unseen occurrence of events in real time based on historical training instances [5]. An example of anomaly detection could be a video surveillance system, in which a number of normal human behaviors such as silhouettes of human bodies, are captured as time series. The system then tries to detect novel and unseen behaviors by looking for the human silhouettes that are different from those time series that are considered normal human behaviors. In this paper, we focus on anomaly detection (AD) of time series. Given a set $S = \{s_i\}$ of training time series which are labelled as normal (or with more detailed class labels), we want to predict whether a set of testing time series $T = \{t_i\}$ is novel based on S. The accuracy of AD is mainly determined by two criteria: the number of novelties that are detected (recall) and the number of falsely detected novelties (precision). This is the classical tradeoff of maximizing recall and maximizing precision.

Common AD algorithms learn models of normalities by fitting models to training instances deemed as normal [5]. Previously unseen instances are tested by measuring their distances (or anomaly score) to the learned models, and a threshold is used for determining novelties [5]. The goal of AD algorithms is to find effective decision boundaries of the normalities that give the best recall and precision for predicted novelties. The decision regions of novelties are treated as the supplementary regions of normalities because the number of novelties in the training data set is zero or quite small and the novelties are widely distributed in the example space.

Many existing solutions of AD algorithms [20], [16], [14] are based on orthogonal feature data. However, effective orthogonal features cannot be easily extracted from time series of complex shapes (e.g., silhouettes of human bodies). Usually, there is much local shifting and scaling variation in both temporal and amplitude dimensions among various instances of shape-based time series [17] and much information of local shapes will be lost when feature extraction techniques are conducted over global time series. Consequently, many distance measures on time series such as dynamic time warping (DTW [3]), edit distance on real sequence (EDR [6]) are computed directly based on the original sequences instead of on extracted features. They have achieved better classification accuracy than many machine learning techniques that are dependent on features extracted from time series [19]. An alternative approach to performing AD is to use existing distance-based outlier detection algorithms [4], [10], [15], in which a testing instance is treated as an instance of S. An anomaly is detected if the testing instance is seen as an outlier within the background of normal training instances in S. However, these algorithms are unsupervised algorithms, where the detailed class labels are not utilized. Moreover, these algorithms are limited to metric distances for efficiency.

In this paper, we propose a novel algorithm for AD of time series which learns effective decision boundaries of normalities from the local neighborhood of training instances. Our solution achieves better accuracy by using a supervised learning technique and some effective distance measures of time series. It improves efficiency by summarizing similar training instances and learning general shapes of time series. The rest of the paper is organized as follows. Section 2 introduces some related work. Section 3 proposes the instance-based AD algorithm. Section 4 gives the VarSpace approach. Section 5 shows the experimental study, and Section 6 gives the conclusions.

II. Related Work

A review of the anomaly detection problem is given in [5]. Solutions for AD can be generally classified into three categories: statistical approaches, neural network based approaches, and nearest neighbor based approaches. Statistical approaches [20] give an explicit estimation of probability distribution of normalities by assuming that the data points follow a family of known distribution. Therefore, certain parameters are learned to fit the distribution in these approaches. Neural network based approaches are also widely used in AD [13] as no explicit probability density model is needed and few parameters need to be set. Like many other machine learning techniques, neural network depends

978-1-4244-6789-1/10/$26.00 ©2010 IEEE

on effective features. Nearest neighbor based approaches [14] identify novel instance based (IB) on the closeness of testing instances to training instances. Some studies [7], [9], [12] on AD of time series have been tried. However, they are all focused on detecting surprising patterns within long streaming time series. In these studies, the shapes of normalities are simple or periodical. Our work is different from these studies in that we detect novel time series instances. One related work of detecting novel time series is to find unusual shapes [18] based on time series. However, it is not focused on the AD problem.

Distance-based learning techniques assume that the closer a testing instance is to a training instance, the higher likelihood that the testing instance has the same class label as the training instance. The traditional kNN classification and distance-based outlier detection approaches are all based on this assumption. Knorr and Ng [10] first gave a definition of distance-based outliers: an instance is predicted as an outlier if it cannot find enough (k) supports within its neighborhood of δ. Ramaswamy et al. [15] proposed an alternative definition of outliers by ranking degrees of outliers calculated from their k^{th} nearest neighbor distance. A global decision strategy is used in these kNN outliers, without considering the variation of local density in the training space. Breunig et al. [4] observed the impact of local density on the effectiveness of outlier detection and proposed a new outlier definition called LOF (local outlier factor), in which the local neighborhood of training instances are used to describe the local density. In LOF, the variation of local density is handled by the relative local reachability. Euclidean distance is very sensitive to distortion and noise within time series [6], [8]. Many efforts have been focused on finding effective distance measures of time series. DTW [8] and EDR [6] have been proposed, and they achieve the best classification accuracy in many time series datasets [19]. The effectiveness of AD can be measured based on the detection rate, false alarm rate, which are used to plot precision-recall curves [11]. They are computed from the number of true positives, false positives, true negatives and false negatives. High detection rate should be guaranteed when AD is applied in surveillance applications.

III. Anomaly Detection of Time Series

Given a training data set S, the label set L contains all class labels of instances in S. Given a training instance $s \in S$, the labels of s form a set denoted as $L(s) \subseteq L$. In most cases, s only has one label, i.e., $|L(s)| = 1$. We however do not exclude the cases that s has multiple labels. In these cases, $|L(s)| > 1$, and those class labels will not be mutually exclusive. An instance $e \in S$ is said to be an enemy of s if $L(e) \cap L(s) = \emptyset$. The set of all the enemies of s in S is denoted as $E(s)$. Correspondingly, an instance $f \in S$ is said to be a friend of s if $L(e) \cap L(s) \neq \emptyset$. The set of all the friends of s in S is denoted as $F(s)$. In the AD problem, a subset of labels $L_M \subseteq L$ is defined as labels of normalities. Correspondingly, the subset of novel labels is denoted as $L_V = \bar{L_M}$. A training instance $s \in S$ is said to be a normal training instance if $L(s) \cap L_M \neq \emptyset$. The set of all normal instances in S is denoted as M. Correspondingly, the set of all novelties in S is denoted as $V = \bar{M}$. In many AD applications, we assume that $|V| \ll |M|$, or even $|V| = 0$.

A. General Distance-based Anomaly Detection

Outlier detection tries to find those outlying instances within a dataset S. The AD problem can be transformed to an outlier detection problem if we simply treat the testing instance t as an instance in S and check whether t is an outlier in the space formed by instances of S. Based on this principle, we extend two distance-based outlier algorithms, kNN outliers [10] and LOF outliers [4], to the AD problem, and propose two general Distance-based Anomaly Detection (DBAD) algorithms. For ease of description, we first give the definition of k-distance of an instance:

Definition 1 (k-distance): Given parameter k, a dataset M of normal training instances, and distance function $d(\cdot,\cdot)$, the k-distance of an instance t, denoted as $dis_k(t)$, is the distance of the k^{th} nearest neighbor (in M) of t to t.

The k-distance measures the closeness of t to the given normal training instances. The lower the $dis_k(t)$, the more evidence of normal training instances can be found around t, and the higher the likelihood of t belonging to a normal instance. Based on kNN outliers given in [10], we define *kNN anomaly* as:

Definition 2 (kNN anomaly): Given parameter k, distance function $d(\cdot,\cdot)$, and a distance threshold δ, an instance t is predicted as a *kNN anomaly* if $dis_k(t) > \delta$; otherwise, it is a *kNN normality*.

In this definition, the novelties are detected when they are found in low density regions, i.e., not enough evidence of normal training instances can be found around them. The parameter δ is user specified. It affects the precision and recall of novelties. Small δ generates high recall of novelties, but decrease the precision. However, similar to the outlier ranking in [15], effective value of δ can be learned from the distribution of $dis_k(s)$, where $s \in S$.

Given $dis_k(t)$, the k-distance neighborhood of t, denoted as $kNN(t)$, contains instances of k nearest neighbors of t in M. Note that multiple instances in M may have the same distance $dis_k(t)$ to t. In this case, to determine the memberships of $kNN(t)$, some other information of instances can be used to break the tie. In LOF [4], given parameter k, distance function $d(\cdot,\cdot)$, and an instance t, the local reachability and local outlier factor of t are defined as: $lrd_k(t) = 1/(\frac{1}{k}\Sigma_{o \in kNN(t)} \max\{dis_k(o), d(t,o)\})$ and $lof_k(t) = \frac{1}{k}\Sigma_{o \in kNN(t)} \frac{lrd_k(o)}{lrd_k(t)}$ respectively. Then we have,

Definition 3 (LOF anomaly): Given parameter k, distance function $d(\cdot,\cdot)$, and a LOF threshold δ, an instance t is predicted as a *LOF anomaly* if $lof_k(t) > \delta$; otherwise, it is a *LOF normality*.

Note that the above two definitions of novelties are independent on distance function $d(\cdot,\cdot)$. When they are applied in AD of time series, various distance measures can all be adopted. We will refer to these algorithms as distance-based

AD algorithms because distances are directly used as distance function $d(\cdot,\cdot)$. However, distance function $d(\cdot,\cdot)$ can also be extended to some other measures describing the difference between instances. Next, we will show how to improve the effectiveness of $d(\cdot,\cdot)$ in AD of time series.

B. Instance-based Anomaly Detection

In our proposed DBAD algorithms, the decision is highly dependent on the distances of k-distance neighborhood since the probability of homogeneity (the same class membership) is directly measured from the distances of k-distance neighborhood instances. However, in many time series applications, distribution of instances are typically skewed. The local instance density of both intra-class instances and inter-class instances may vary significantly. As a result, the simple distance may not be accurate enough to measure the probability of instance homogeneity. An example in a two dimensional space is shown in Fig. 1.

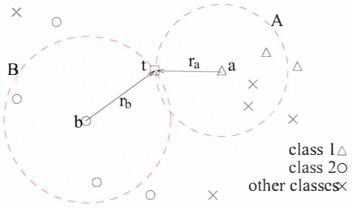


Figure 1. A case where DBAD is not effective

In this example, t is a test instance, a and b are 2 nearest neighbors of t. Suppose only the nearest neighbor is used in the DBAD. t is most likely to be predicted as a friend of a since a is close enough to t. However, for a, t is not close enough to a since there are some enemies (some of them may be novelties) of a who are closer than t to a. Therefore, a false affiliation may occur if t is simply predicted as a friend of a. For example, instance b is further away from t than a, it will be ignored since only $1NN$ is used. However, for b, t is close enough since there is no enemy of b whose distance to b is less than r_b, and there are some friends of b whose distances to b are larger than r_b. Therefore, compared to a, b is most likely to be an evidence of t. This case may happen when the local instance density of b is much lower than that of a.

Both the class labels of normal training instances and the relative distribution of friends and enemies are ignored in the DBAD algorithms despite the fact that they may help to constrain possible extension of instances within one normal class and therefore are useful in building effective decision boundaries of normal instances. To explore the benefits of detailed class labels, we propose an instance-based AD (IBAD) algorithm. We re-design the basic distance function $d(\cdot,\cdot)$ by reducing it to a non-symmetric distance measure which takes into account the relative distribution of friends and enemies around the reference instances. Therefore, in IBAD, $d(t,s)$ measures the closeness of t to s from the view point of s which may not be equal to $d(s,t)$.

On the analogy of the definition of k-distance neighborhood, we can define the k-support neighborhood of a training instance $s \in M$ as, $kNN_f(s)$, which contains instances of k nearest neighbors of s in $F(s)$. Similarly, the k-fence neighborhood of a training instance $s \in M$ is $kNN_e(s)$, which contains instances of k nearest neighbors of s in $E(s)$.

Definition 4 (k-support): Given parameter k, distance function $D(\cdot,\cdot)$, and a normal training instance $s \in M$, the k-support of s is defined as $sup_k(s) = \frac{1}{k}\sum_{o\in kNN_f(s)} D(o,s)$.

Definition 5 (k-fence): Given parameter k, distance function $D(\cdot,\cdot)$, and a normal training instance $s \in M$, the k-fence of s is defined as $fen_k(s) = \frac{1}{k}\sum_{o\in kNN_e(s)} D(o,s)$.

In these two definitions, function $D(\cdot,\cdot)$ is an original distance measure of time series. The k-support and k-fence of a training instance s are the average distances of k-support neighborhood and k-fence neighborhood of s. They measure how far the friends and enemies of s are distributed around s. The closeness of a testing instance t to s can be defined based on the relative distribution of t to the k-support and k-fence of a training instance s:

Definition 6 (novel score): Given parameter k, distance function $D(\cdot,\cdot)$, an instance t and a normal training instance $s \in M$, the novel score of t to s is defined as: $IB_k(t,s) = \frac{D^2(t,s)}{fen_k(s)\cdot\min(sup_k(s),fen_k(s))}$.

The purpose of proposing novel scores is to measure the closeness of similar instances more accurately, compared to the direct distance measure $D(t,s)$. Note that the local density variation handled by LOF is handled in novel scores based on the relative distance distribution of $D(t,s)$ to $sup_k(s)$ and $fen_k(s)$. Moreover, the relative distribution k-support and k-fence of s ($\min(sup_k(s), fen_k(s))$) is also considered in the calculation of novel scores. An example is shown in Fig. 2, where t will be closer to s in case (a).

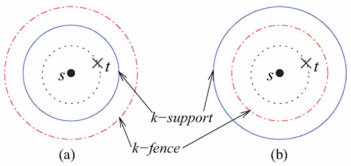


Figure 2. Relative distribution of k-support and k-fence. It is guaranteed that $IB_k(t,s)_a < IB_k(t,s)_b$.

The computation of novel scores is a kind of local supervised learning process since local friends and enemies of training instances are used for computing the novel scores. By replacing the basic distance function $d(\cdot,\cdot)$ in Definition 2 with $IB_k(\cdot,\cdot)$, we get the IBAD algorithm:

Definition 7: Given parameter k and k_1, distance function $d(\cdot,\cdot) = IB_{k_1}(\cdot,\cdot)$, and a distance threshold δ, an instance t is predicted as a *IB anomaly* if $dis_k(t) > \delta$; otherwise, it is a *IB normality*.

IV. The VarSpace Approach

A. Anomaly Detection with LB Supports

The above DBAD and IBAD algorithms are independent of applications and distance measures. When they are applied in AD of time series, our experiments (in section V) show that the combination of IBAD and warping distances such as

DTW and EDR achieves much better accuracy than the other combination. Note that compared to Euclidean distance, the computational cost of DTW and EDR is much expensive. To improve the efficiency of IBAD, lower bounding techniques such as LB_Keogh [8] can be applied to avoid actual distance computation of time series. Since these lower bounds are designed for DTW distance, the following studies on efficiency are mainly focused on DTW distance.

Given a test time series t, a naive way of IBAD is to compute the novel score of t to every training instance $s \in S$. An evidence is found when the anomaly score is less than a threshold δ. The algorithm terminates when k evidences are found, and t will be predicted as a *IB normality*. Otherwise, it is a *IB anomaly*. Since each novel score requires one actual distance computation, the computational cost will be high when t is compared with a large number of training instances. LB_Keogh can be employed to provide lower bounds of $IB_k(\cdot,\cdot)$ so that some actual distance calculation can be avoided.

B. VarSpace: Summarize Training Instances

Although LB_Keogh can dramatically reduce the number of actual distance computation, due to the looseness of LB_Keogh, the remaining distance computation is still significant (more than the cost of computing lower bounds in our tests). We observe that there is much computational redundancy in IBAD because similar novel scores are usually obtained when t is compared to similar instances. In the training dataset, distribution of instances is usually highly skewed. Some normal instances are well clustered into a bunch of clusters, which represent the centers of normal class memberships. If similar instances within a cluster can be summarized as one instance, the computational cost will be saved as the test instance only need to compare with the summarized instance once, instead of comparing with every member in that cluster. Inspired by this, we propose a local instance summarization approach, called VarSpace.

We propose to generalize the shape variation of an instance s and its close friends by using the convex of s, denoted as $con(s)$. $con(s)$ is computed from broadening the value range of each data item within s. It can be represented as a series of ranges, $con^k(s) = [(L_1,U_1),\ldots,(L_n,U_n)]$. $con^k(s)[i] = (L_i,U_i)$ is the lower bound and upper bound of the convex at position i. It represents the shape variation of s in the local position of i. k is the number of neighbor friends of s for learning $con(s)$. Given a test instance t and a convex $con^k(s)$, the distance of t to $con^k(s)$, denoted as $D'(t, con^k(s))$, is computed by treating $s[i]$ as a wildcard value of range (L_i,U_i). During the process of distance computation, an optimized value of $s[i]$ is chosen from (L_i,U_i) such that the local distance (at position i of s) can be minimized.

In the VarSpace approach, $con^k(s)$ is a summarization of an instance s and its k-support neighborhood. A simple way of learning $con^k(s)$ is the exact shape convex of s and its k-support neighborhood. We call this convex the simple convex, which can be defined as: $con^k(s).L_i = \min(s[i], \min_{o\in kNN_f(s)} o[i])$, $con^k(s).U_i = \max(s[i], \max_{o\in kNN_f(s)} o[i])$, $i = 1,\ldots,n$. Fig 3 shows an example of simple convex learned from some similar time series.

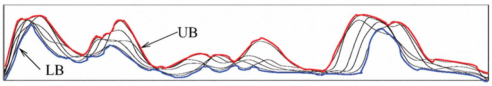


Figure 3. Simple convex learning

Simple convex learning is quite straight-forward. However, due to some local time shifting among similar instances, the convex obtained is usually too loose, which result in loss of accuracy because the shapes of some enemies of s can also easily fall in the convex of s. An inaccurate convex representation can significantly reduce the distances of friends and enemies of s. To make the convex representation tighter, we propose another type of convex, called flexible convex.

DTW distances are computed using dynamic programming. The distances are aggregated within a distance matrix. The warping path of a DTW distance can be obtained by tracing from the top right corner back to the bottom left corner of the distance matrix. It can be described as a sequence of $(s_i,e_i), i = 1,\ldots,n$. s_i and e_i are the start position and the end position (vertically) of warping path at the position i (horizontally). Based on the warping paths of s' k-support neighborhood, the local time shifting of k-support neighborhood of s can be removed by aligning shapes along the warping paths, so that the difference of corresponding data items between s and its k-support neighborhood can be reduced. The flexible convex of s can be computed by Algorithm 1. Fig. 4 shows the flexible convex of s in the running example. It is obvious that the flexible convex gives a tighter bound than the simple convex. Another phenomena in flexible convex is that inaccurate regions of the convex are typically located in regions of local peaks and valleys of s.

Algorithm 1 Flexible convex learning

1. **Input**: S, a set of training instances, $s \in S$
2. **Input**: k, the number of neighbor friends of s
3. **Output**: $con^k(s)$, the convex of s
4. n = GetLength(s)
5. **for** $i = 1$ to n **do**
6. $\quad con^k(s).L_i = s[i]$
7. $\quad con^k(s).U_i = s[i]$
8. **for** every instance $o \in kNN_f(s)$ **do**
9. $\quad$ GetDTWWarpingPath(o, s)
10. $\quad$ **for** $i = 1$ to n **do**
11. $\quad\quad$ **for** $j = s_i$ to e_i **do**
12. $\quad\quad\quad$ **if** $o[j] < con^k(s).L_i$ **then**
13. $\quad\quad\quad\quad con^k(s).L_i = o[j]$
14. $\quad\quad\quad$ **if** $o[j] > con^k(s).U_i$ **then**
15. $\quad\quad\quad\quad con^k(s).U_i = o[j]$

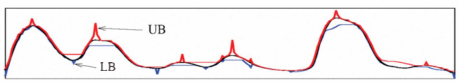


Figure 4. Flexible convex learning

In the VarSpace approach, an instance s and its k-support neighborhood is summarized as $con^k(s)$. To summarize the whole training instance space, all instances $s \in S$ are ranked by reverse density, $r(s) = \frac{sup_k(s)}{fen_k(s)}$ which measures the closeness of similar instances to s. The instance s with the lowest

$r(s)$ is first chosen as a cluster center. s and its k'-support neighborhood (k' is chosen such that $\frac{dis_{k'}(s)}{fen_k(s)}$ is lower than a given threshold of reverse density ξ) are then summarized as $con^{k'}(s)$, and removed from the candidate list of new cluster centers. The algorithm continuously chooses a new cluster center with the lowest reverse density in the candidate list until the reverse density of candidates exceeds ξ. Finally, a number of convexes will be extracted from S.

C. Anomaly Detection with VarSpace

The novel score of an instance to a convex is defined as Definition 8. In this definition, the $sup_k(c)$ and $fen_k(c)$ are measured based on $D'(\cdot,\cdot)$.

Definition 8: Given parameter k, distance function $D'(\cdot,\cdot)$, an instance t, and a convex $c = con^{k'}(s)$, the novel score of t to c is defined as $IB_k(t,c) = \frac{D'^2(t,c)}{fen_k(c)\cdot\min(sup_k(c),fen_k(c))}$.

The LB_Keogh still exists for the DTW of an instance t and a convex $con^k(s)$. When computing the upper envelope [8] of $con^k(s)$, $con^k(s).U$ is used to replace the original series s. Correspondingly, when computing the lower envelope of $con^k(s)$, $con^k(s).L$ is used to replace the original series s. Such a modification guarantees that the modified lower bounds satisfy the inequality, $LB\_Keogh(t, con^k(s)) \leq DTW(t, con^k(s))$. By applying the novel scores of convexes, we develop the IBAD algorithm with VarSpace support in Algorithm 2.

Algorithm 2 IBAD with VarSpace supports

```
1. Input: t, test time series
2. Input: S', a mix set of training instances and convexes
3. Output: is_novel, whether t is novel or not
4. n' = |S'| /*size of summarized training data set*/
5. cand_list.initialize(); num_cand = 0
6. for i = 1 to n' do
7.    s = S[i] /*a training instance or a convex*/
8.    dis = LB_Keogh_IB(t, s)
9.    if dis ≤ δ then
10.      if (s is a convex) then
11.         num_cand+ = con(s).size()
12.      else
13.         num_cand + +
14.      node.ins = s; node.dis = dis; cand_list.push(node)
15. if num_cand < k then
16.    Return is_novel = true
17. else
18.    evidence = 0
19.    while cand_list.size() > 0 do
20.       s = cand_list.popInstanceOfMinDis()
21.       if IB_k1(t, s) ≤ δ then
22.          if (s is a convex) then
23.             evidence+ = con(s).size()
24.          else
25.             evidence + +
26.          if evidence ≥ k then
27.             Return is_novel = false
28.    Return is_novel = true
```

V. Experiment

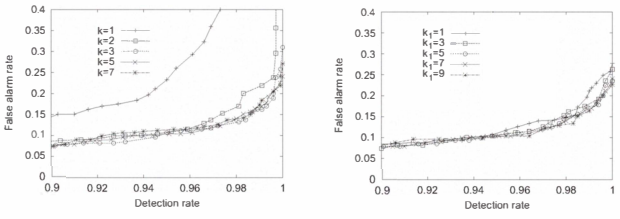
A. Data Sets and Effectiveness of AD

The time series datasets we use include 50words and OSULeaf datasets from UCR time series classification/clustering page [1]. We also tested over some other datasets which have similar phenomena as the reported two datasets. They are not shown due to the page limit. There are hundreds of instances in each dataset. 10-fold cross validation is used to test the accuracy. In each test, around 90% of the normal instances are used as training instances. The others are served as test instances. The accuracy is computed over all the results from the ten tests. Unlike the common way of drawing precision-recall curves [11], we propose to represent the effectiveness of AD algorithms as Fig. 5. We focus on regions of high detection rates. The ideal curve is the bottom line with zero false alarm rate.

B. Performance of IB anomaly

To study the performance of IBAD, we compare the efficacy of three approaches (*kNN anomaly*, *LOF anomaly* and *IB anomaly*) under three distance measures respectively. The results are shown in Fig. 5. The results in these tests clearly show that the efficacy of *IB anomaly* are all better than that of *kNN anomaly* and *LOF anomaly*, regardless of the applied distance measures and data sets.

We adjust the parameter k in *kNN anomaly* and *LOF anomaly* algorithms. The values of k with the best accuracy are used in Fig. 5. For *kNN anomaly*, small value of k ($k \leq 3$ in various distance measures) achieves the best accuracy. While, the best k in *LOF anomaly* is relatively larger ($k \geq 7$) than that in *kNN anomaly* since the local density depends on the average distance of certain number of neighbors. For *IB anomaly*, the best k is moderate. We use one test case to show the impact of k and k_1 on the accuracy of *IB anomaly* under DTW distance in Fig.6. We can see that the efficacy of *IB anomaly* is very stable when k and k_1 are larger than 3. In our tests, we simply choose $k = 5$ and $k_1 = 7$.



(a) Parameter k (b) Parameter k_1

Figure 6. Impact of parameters on IBAD.

C. Performance of The VarSpace Approach

To show the performance of VarSpace approach, we compare IBAD with VarSpace supports (Algorithm 2) with the IBAD with lower bound supports ($IBAD_{LB}$). The results are shown in Fig. 7. We compare the false alarm rate and computational cost. For the VarSpace approach, the threshold of reverse density (ξ) is adjusted so that various scales of instance summarization can be conducted. The results clearly show that when ξ is enlarged, the computational cost can be reduced to less than 1/6 of the $IBAD_{LB}$ ($\xi = 0$). While the accuracy of AD (in terms of false alarm rate) is not affected too much by the VarSpace approach. The false alarm rates of some tests are even lower than those of original IBAD algorithm. This experimental study shows that we may reduce

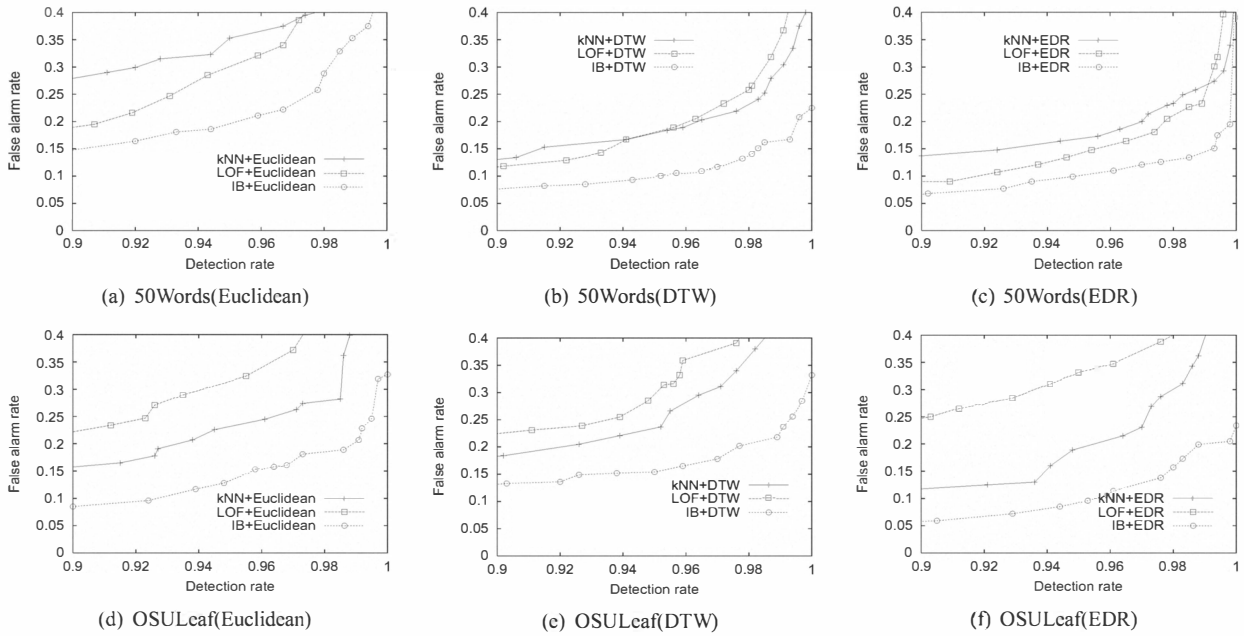


Figure 5. Efficacy comparison of AD algorithms.

the size of training instance space and improve the efficiency of AD by enlarging the threshold of reverse density.

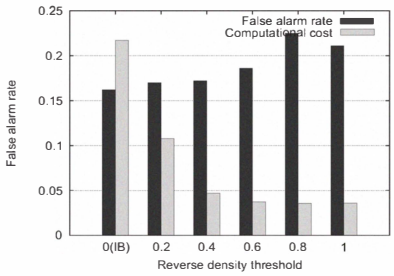


Figure 7. Performance of AD with VarSpace (50words, Detection Rate = 0.956)

VI. Conclusion

We propose a nearest neighbor approach by extending the existing distance-based outlier detection algorithms to the problem AD of time series. To utilize the detailed class labels of training data set and build more effective decision boundaries of normal instances, we propose the IBAD algorithm, in which the distances are revised as novel scores obtained from local neighborhood of training instances. To improve the efficiency of AD, we propose the VarSpace approach, which summarizes similar training instances by learning the local shape variation of similar instances. The experiments show that the IBAD outperforms the DBAD algorithms in terms of accuracy significantly. When the IBAD algorithm runs on the summarized training instance space, the efficiency can be improved significantly without the loss of accuracy too much.

References

[1] *UCR Time Series Page*. http://www.cs.ucr.edu/∼eamonn/time_series_data/.

[2] Faisal I. Bashir, Ashfaq A. Khokhar, and Dan Schonfeld. Real-time motion trajectory-based indexing and retrieval of video sequences. *IEEE Transactions on Multimedia*, 9(1):58–65, 2007.

[3] Donald J. Bemdt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD Workshop*, pages 359–370, 1994.

[4] M.M. Breunig, H.P. Kriegel, R.T. Ng, and J. Sander. Lof: Identifying densitybased local outliers. In *ACM SIGMOD*, pages 427–438, 2000.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):1–58, 2009.

[6] Lei Chen, M. Tamer Ozsu, and Vincent Oria. Robust and fast similarity search for moving object trajectories. In *ACM SIGMOD*, pages 491–502, 2005.

[7] D. Dasgupta and S. Forrest. Novelty detection in time series data using ideas from immunology. In *Proceedings of The International Conference on Intelligent Systems*, pages 19–21, 1996.

[8] Eamonn Keogh. Exact indexing of dynamic time warping. In *VLDB*, pages 406–417, 2002.

[9] Eamonn Keogh, Stefano Lonardi, and Bill Chiu. Finding surprising patterns in a time series database in linear time and space. In *ACM SIGKDD*, pages 550–556, 2002.

[10] E. Knorr and R. Ng. Algorithms for mining distance based outliers in large data sets. In *VLDB*, pages 392–403, 1998.

[11] Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detections. In *ACM SIGKDD*, pages 157–166, 2005.

[12] Junshui Ma and Simon Perkins. Online novelty detection on temporal sequences. In *ACM SIGKDD*, pages 417–423, 2003.

[13] A.F. Murray. Novelty detection using products of simple expertsła potential architecture for embedded systems. *Neural Networks*, 14:1257–1264, 2001.

[14] Dragoljub Pokrajac, Aleksandar Lazarevic, and Longin Jan Latecki. Incremental local outlier detection for data streams. In *CIDM*, pages 504–515, 2007.

[15] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD*, pages 427–438, 2000.

[16] Guojie Song, Bin Cui, Baihua Zheng, Kunqing Xie, and Dongqing Yang. Accelerating sequence searching: dimensionality reduction method. *Knowl. Inf. Syst.*, 20(3):301–322, 2009.

[17] Ashok Veeraraghavan, Rama Chellappa, and Amit K. Roy-Chowdhury. The function space of an activity. In *CVPR*, pages 959–968, 2006.

[18] Li Wei, Eamonn Keogh, and Xiaopeng Xi. Saxually explict images: Finding unusual shapes. In *ICDM*, pages 711–720, 2006.

[19] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana. Fast time series classification using numerosity reduction. In *ICML*, pages 1033–1040, 2006.

[20] Dit Yan Yeung and Y. Ding. Host-based intrusion detection using dynamic and static behavioral models. *Pattern Recognition*, 36(1):229–243, 2003.