Detection and Implicit Classification of Outliers via Different Feature Sets in Polygonal Chains

Michael Singhof[1], Gerhard Klassen[2], Daniel Braun[1], Stefan Conrad[1]

Abstract: Many outlier detection tasks involve a classification of outliers of different types. Most standard procedures solve this problem in two steps: First, an outlier detection algorithm is carried out, which is normally trained on outlier free data, only, since the samples of outliers are limited. Second, the outliers detected in that step, are classified with a conventional classification algorithm, that needs samples for all classes. However, often the quality of the classification is lowered due to the small number of available samples.

Therefore, in this work, we introduce an outlier detection and classification algorithm, that does not depend on training data for the classification process. Instead, we assume, that different kinds of outliers are inferred by different processes and as such should be detected by different outlier detection approaches. This work focuses on the example of outliers in mountain silhouettes.

Keywords: Anomaly & Outlier Detection, Classification, Image Segmentation

1 Introduction

The detection of outliers in a data set occurs in many contexts, ranging from clustering algorithms such as DBSCAN [Es96] to credit card fraud detection [CS98, Ch99], function tests of aircraft engines [Ab16], or the detection of cyber attacks [La04, CBG12] and many other application areas. In some cases, like clustering, it is sufficient to just erase any point that is either noise or an anomaly. In other cases, like function tests, the outliers are of primary interest. A general problem in those cases is the fact, that normally, none or very few outlier instances are known. Therefore, outlier detection is mostly treated as a single class problem, where for each data point it is rated, whether that point is normal or not. If it is not judged as being normal, it is treated as an outlier.

In some cases, however, a further classification of outliers is necessary. In [BSC16], we presented a system that is able to find a mountain's silhouette in a photo. It utilises an outlier detection algorithm to get rid of errors during the segmentation step, that belong to different classes. In this case, it is important to differentiate between these classes, because errors of different kinds get corrected in different ways.

The remainder of this paper is structured as follows: Chapter 2 explains and motivates the problem we try to solve, chapter 3 gives an overview of related work. In chapter 4 we describe

[1] Heinrich-Heine-Universität Düsseldorf, Institut für Informatik, Universitätsstraße 1, 40225 Düsseldorf, {singhof, braun,conrad}@cs.uni-duesseldorf.de

[2] Heinrich-Heine-Universität Düsseldorf, Institut für Informatik, Universitätsstraße 1, 40225 Düsseldorf, gerhard.klassen@uni-duesseldorf.de

our approach to outlier detection and how it can be used for an implicit classification. We evaluate our approach in chapter 5 and finally draw a conclusion in chapter 6.

2 Motivation and Problem Description

In this work, we present an approach to outlier detection and classification on polygonal chains, that are given by an image segmentation algorithm. The aim of this framework as a whole is the automatic annotation of mountain photos, by detecting the silhouette of a mountain in a given image and then comparing it to a set of reference silhouettes.

During the segmentation step several problems can occur that might obscure the silhouette: First, there can be obstacles in the photo that are in front of the mountain, such as trees, buildings or persons. In order to extract an exact silhouette it is necessary to take note of such obstacles and ignore them in the silhouette matching step. Second, segmentation errors can occur that can by caused by a low contrast between sky and foreground. This happens if clouds occur close or overlapping to the silhouette, if snowfields appear next to light sections of the sky or for other reasons where contrast between sky and foreground is minor. Some of these errors are shown in figure 1, that has obstacles in the form of trees on the left side and segmentation errors due to low contrast on the right hand side.

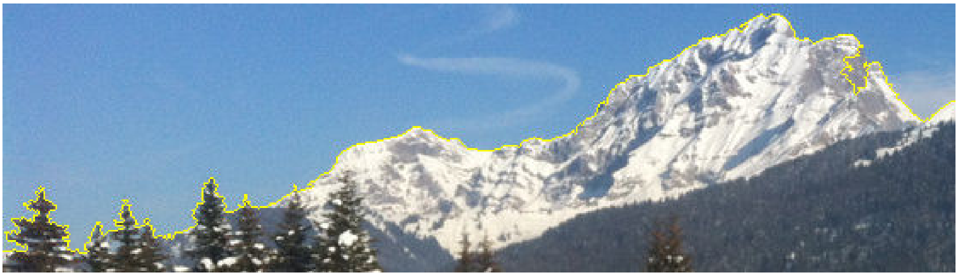

Fig. 1: Examples for different errors in a mountain silhouette.

Since we use an adaptive segmentation algorithm, it is possible to correct errors if we detect them. Figure 2 gives an overview of the architecture of the segmentation module. In general, a grid is laid over the image. For the initial segmentation, the same parameters are used for every cell, although these can be changed during the adaptive process. Then, the segmentation algorithm computes a silhouette from the image, which is passed to the outlier detection step. When no outliers are found, the silhouette is inferred as clean and the algorithm terminates. In the case that outliers are found, these are passed to the classification module. If an outlier gets classified as being an obstacle, it is removed by replacing it by a straight line. Otherwise, if an outlier is classified as a segmentation error, it is passed to the segmentation module. There, the parameters for the affected grid cells are changed to better accommodate the local circumstances.

This work concentrates on the detection and classification of outliers. A general problem with outlier detection, as mentioned in the introduction, is the fact, that in most cases no exhaustive collection of examples for all shapes of outliers exists. This is the case with

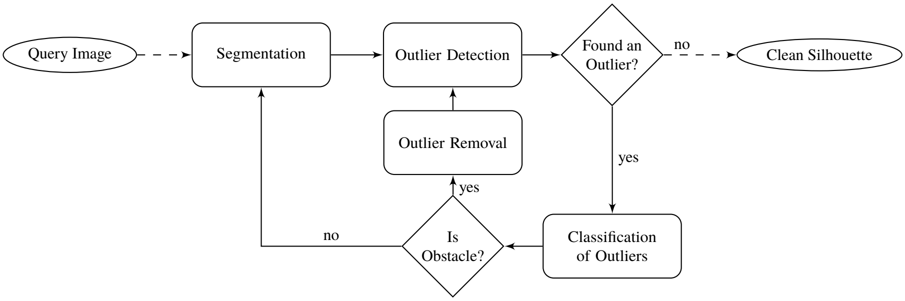

Fig. 2: Flow diagram of adaptive image segmentation.

our problem, too. Therefore, our outlier detection algorithm is trained on normal data, only. However, for a regular classification samples of all classes are needed. One can argue here, that for most outliers that are found, a small collection of examples is sufficient to differentiate between the given classes. In previous work [BSC16], we have used such a solution to acceptable results.

In our current approach, that has been outlined in [SBC16], we only use the geometric properties of the silhouette to depict outliers. In this work, however, we want to implicitly classify the detected outliers by using different sets of features for the different types of outliers. Our notion is, that those different outliers are produced by different mechanisms and thus can be detected by looking at different attributes: On one hand, segmentation errors occur in regions with unusual low contrast along the silhouette. On the other hand, obstacles usually have borders to the sky that are as sharp as the rest of the silhouette in the surroundings of the obstacle but have unusual forms for mountain silhouettes.

3 Related Work

There is an abundance of work on outlier detection, beginning with statistical models [Ha80, BL94] and general definitions of outliers [BC83] to very specialised applications of outlier detection. Some of them have been already mentioned in the introduction such as [CS98, La04, CBG12, Ab16]. Since, to our knowledge, there is no work on outlier detection on polygonal chains in general, the field that is closest related to ours is outlier detection on time series. There are two major problems for outliers in time series, namely the finding of change points and the finding of unusual subseries. A change point is a certain point in time, where the time series changes its behaviour drastically. This has, among others, researched in [FP99, KS09]. Well known approaches to the problem of finding unusual parts of time series include HotSax[KLF05] and specialisations of it such as [PLD10, BA11b, KA12].

The basic idea of HotSax is finding the strongest discord. A discord is a subsequence of a time series, that does not fit the general shape of the time series it lies in. HotSax computes the one most unusual part of a time series consisting of k points where k is given by the user. In contrast to this, for a given polygonal chain, the target of this work is to find all outliers of arbitrary lengths for a given polygonal chain, including the possibility of not finding an outlier at all if there is none in the data.

The target of mountain recognition was first tackled in [Ba12]. Together with this paper, a corpus of 203 annotated images was released, which is often used in this field. In contrast to our approach, the approach by Baatz et al. requires human intervention in some cases. Kim et al. [Ki11] introduced a skyline detection algorithm that uses a Canny edge detection [Ca86],first, and then filters the resulting edges in order to get silhouette edges, only. One disadvantage of this algorithm is the fact, that it does not find a continuous silhouette but in most cases only parts of it. The authors of [Ah15] use a combination of different techniques, both edge-less and edge-based in order to come up with a skyline. Baboud et al. [Ba11a] use a similar technique in order to annotate mountains, that relies on GPS coordinates and does not extract a skyline or silhouette in particular.

4 Outlier Detection and Classification

As mentioned in section 2, the segmentation part of our framework passes a silhouette in the form of a polygonal chain to the outlier detection part. This gets searched for untypical parts and then those parts are given to the classification module. In this section, we describe the current approach two-dimensional approach and introduce changes to the work presented in [BSC16, SBC16], namely the addition of further dimensions and the proposal of merging strategies.

To understand the nature of outliers, we first have to introduce the construct of the silhouette. Visually speaking, the silhouette is the border in the image, that separates the sky from the foreground, or in our case, that separates the mountain from everything above the mountain, that might include objects in front of the mountain.

Formally, we define a silhouette as follows:

Definition 1. Let $S = (p_1, \ldots, p_n)$ be a sequence of points $p_i = (x_i, y_i) \in [0, x_{\max}]\times[0, y_{\max}]$ for an image of the size of $x_{\max} \times y_{\max}$ pixels. S is called a *silhouette* if the points p_1 and p_n lie on the borders of the image.

Given a silhouette S, an outlier is a sub-sequences $O = (p_i, \ldots, p_j)$, $1 \leq i < j \leq n$, that marks an unusual part of the silhouette. This unusualness is expressed by an anomaly score on single points of a silhouette. The bigger that anomaly score is, the more unusual a point is. An outlier consists of a series of points that each have a high anomaly score.

4.1 Outlier Detection

After the previous section introduced a general idea of the term outlier, in this section we describe how outliers are computed. Figure 3 gives an overview over the outlier detection process.

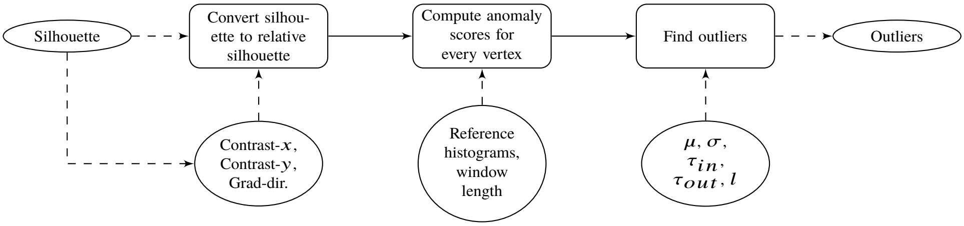
Silhouette
Convert silhouette to relative silhouette
Compute anomaly scores for every vertex
Find outliers
Outliers
Contrast-x, Contrast-y, Grad-dir.
Reference histograms, window length
μ, σ, τ_{in}, τ_{out}, l

Fig. 3: Flow diagram of outlier detection.

It can be seen there, that the process is essentially split in three steps; the first of which is the conversion of a silhouette S to a so-called relative silhouette RS. In contrast to a silhouette, in a relative silhouette we give the coordinates of every vertex relatively to the coordinates of its predecessor in the silhouette. In our case we chose a polar coordinate like notation. We also add additional information to the silhouette, namely contrast values in x and y direction, as well the gradient direction:

Definition 2. Let $S = (p_1, \ldots, p_n)$ be a silhouette. $RS = (v_1, \ldots, v_n)$ is called the *relative silhouette of* S if $v_1 = (0, 0, cx_1, cy_1, g_1)$ and for each $v_i = (sl_i, a_i, cx_i, cy_i, g_i)$ for $i \in [2, n]$, $sl_i = |p_i - p_{i-1}|$ denotes the length of the the line segment between p_i and p_{i-1}, a_i denotes the angle between that line segment and the x-axis, cx_i and cy_i give the contrast at the point p_i in direction of the x, respectively y axis, and g_i denotes the contrast direction at point p_i.

The computation of the relative silhouette for a given silhouette is straight forward and can be carried out in $O(n)$, for n the number of points in the silhouette S. This is done in order to ensure, that similar structures in the silhouette that lie in different parts of the image are represented in a similar fashion, without the need to do any further computations.

The second step in the process is the computation of the anomaly scores of the single vertices of the relative silhouette RS. As can be seen in figure 3, in this step, additional information is needed, namely reference histograms and a parameter called window length. A relative silhouette – or a part of one – can be easily transformed into a histogram. The histogram's bins consist of up to five dimensions, depending on the chosen features. The reference histograms are histograms derived from error free silhouettes. More on their computation can be found in [SBC16] for the two dimensional case. Computation for other numbers of dimensions are analogous. Let us assume that we have k reference histograms $H^1_{ref}, \ldots, H^k_{ref}$. Then we use a sliding window with the size given by the parameter window size. For every window we compute the corresponding histogram H and with that the distance to the reference histograms as $d = \min_{1 \le i \le k} \operatorname{dist}(H, H^i_{ref})$, for any histogram distance function dist.

We store d for every vertex that has been part of the sliding window so that every vertex v_i gets a collection D_i of distance values.

Definition 3. Let D_i be the collection of distance values for a point v_i. Then we call $an(v_i) = \frac{1}{|D_i|} \sum_{d \in D_i} d$ the *anomaly score* of v_i.

As with the silhouette conversion, it is obvious, that the anomaly scores can be computed in $\mathcal{O}(n)$, since the number of bins of the histograms and therefore the histogram distance function are independent of the number of points in the silhouette.

The third and last step is the actual outlier detection. Again, as shown in figure 3, some additional parameters are necessary. Of these, μ is the mean of the anomaly score distribution computed on the reference data and σ is the standard deviation. These are computed together with the reference histograms from the previous step. In contrast to this, τ_{in}, τ_{out} and l are parameters given by the user. Here, τ_{in} and τ_{out} are thresholds for the anomaly scores and l is the minimal number of points that an outlier has to consist of.

Definition 4. Let $RS = (v_1, \dots, v_n)$ be the relative silhouette of an image with corresponding anomaly scores $an(v_i)$ for vertex v_i, reference anomaly score distribution mean μ and standard deviation σ and two thresholds $0 < \tau_{out} < \tau_{in}$.

Then we call v_i a *weak anomaly* if $an(v_i) \geq \mu + \tau_{out} \cdot \sigma$ and a *strong anomaly* if $an(v_i) \geq \mu + \tau_{in} \cdot \sigma$.

We use a double threshold technique since our observations show that on one hand, if we chose only one relatively high threshold, the detected outliers would often be too small. On the other hand, if we chose a single low threshold, we would find many false positives. Therefore, due to the double thresholds, we can restrain the number of detected outliers by τ_{in} but are able to expand those outlier by choosing a lower value for τ_{out}.

We can now define an outlier in our context as given in [BSC16]:

Definition 5. Let $l > 0$, and $RS = (v_1, \dots, v_n)$ be a relative silhouette. We call $o = (v_i, \dots, v_j)$ an *l-outlier* if the following is true:

1. For all v_k, $i \leq k \leq j$, it holds that v_k is a weak anomaly.
2. There exist $m_1, m_2 \in \{i, \dots, j\}$ such that $m_2 - m_1 \geq l$ and for all v_k, $m_1 \leq k \leq m_2$, it holds that v_k is a strong anomaly.

An outlier $o = (v_i, \dots, v_j)$ is called a *maximum l-outlier* if and only if neither $(v_{i-1}, \dots, v_j)$ nor $(v_i, \dots, v_{j+1})$ are l-outliers.

Our goal is to find all maximum outliers in the silhouette. This is done by iterating over all vertices in the silhouette and then react as noted in table 1. There, s_{out} is a variable that stores the position of a possible start of the outer part of an outlier, or nf if no start has been found yet, s_{in} stores the possible start of an inner part of an outlier, and e_{in} stores the end of an inner outlier. The minimum inner length is given by l and v_i denotes the current vertex.

This can be done in linear time as well, so that the whole outlier detection algorithm can be executed in $\mathcal{O}(n)$.

Status of variables			Status of current vertex v_i		
s_{out}	s_{in}	e_{in}	no anomaly	weak anomaly	strong anomaly
nf	nf	nf	—	set $s_{out} = i$	set $s_{in} = s_{out} = i$
found	nf	nf	set $s_{out} = nf$	—	set $s_{in} = i$
found	$i - s_{in} \leq l$	nf	set $s_{out} = s_{in} = nf$	set $s_{in} = nf$	—
found	$i - s_{in} > l$	nf	set $o_{in} = i$, save o	set $e_{in} = i$	—
found	$i - s_{in} > l$	found	save o	—	—

Tab. 1: Decision matrix for outlier detection.

Number of features	Used features	Outlier type
2	Length, Angle	Obstacle
3	Contrast x, Contrast y, Gradient dir.	Segmentation
5	Length, Angle, Contrast x, Contrast y, Gradient dir.	Segmentation

Tab. 2: Used feature set and their implication on outlier types.

4.2 Merging and Classification of Outliers

If only one feature set is used for the outlier detection, overlapping outliers do not occur, since an outlier is expanded first. That means, we enlarge an outlier, first, before we proceed with searching for the next outlier in the next part of the silhouette. However, if we carry out the outlier detection for different feature sets, in order to use an implicit classification, it is possible for outliers to overlap. We then have to decide, how to deal with those outliers and which class they should have.

The basic assumption for the classification is, that the feature set in which an outlier occurs, indicates the type of outlier. As described in section 2, outliers in the contrast features hint at segmentation errors while outliers with normal contrast but unusual shape hint at obstacles. Table 2 gives an overview of the feature sets we use and what they mean for the outlier types.

We have developed three strategies for the merging of outliers. The first is called “Merge” and, for a set of overlapping outliers creates an outlier that begins at the lowest start index and ends at the highest end index. The type is then computed as the type with the longest inner outlier, i.e. the part of the outlier, that consists of strong anomalies, of the involved outliers. The second strategy, “Merge to Segmentation”, depicts the merged outlier as being a segmentation error if at least one outlier from a segmentation error feature set is involved. Finally, “Split and Merge” is the most complex strategy. Here we look, if the inner parts of outliers intersect. If this is the case, we merge the outliers with the “Merge” strategy. Otherwise, we split the combined outlier at the middle between the inner parts, if both outer outliers are reaching over the middle. Otherwise, the split is performed as close to the middle as possible. The single parts of the outlier then get merged and classified by the “Merge” strategy.

5 Evaluation

The evaluation is carried out on a data set, which consists of 3580 outlier vertices forming 114 outliers that have been manually marked in 14 silhouettes. The silhouettes are automatically detected by the segmentation part of AdaMS, that is described in [BSC16], although without using the adaptive improvement. This is in order to ensure typical outliers for our application scenario. All outlier detection algorithm variants have been trained on 48 mostly outlier free silhouettes. Clustering has been carried out 1000 times and the clustering with the lowest quadratic distances has been chosen, and the number of clusters and reference histograms k has been set to 30.

Method	Found outliers of length						Prec.	Recall	F_1
	$(0,5]$	$(5,10]$	$(10,20]$	$(10,50]$	> 50	Total			
2d	6	21	12	19	14	72	70	70	70
3d	4	11	11	10	14	50	57	66	61
5d	15	36	22	21	15	109	52	81	66
Combined3	8	26	19	19	15	87	58	88	73
Combined5	15	38	23	21	15	112	51	88	70

Tab. 3: Comparison of single and combined approaches.

Table 3 shows the detection results for the single feature sets and the feasible combinations. It can be seen here, that the two dimensional method yields the best results in respect to precision and F_1 measure, while the five dimensional approach finds the most outliers and has the highest recall of the single feature sets. The three dimensional approach has the worst results, however this is as expected, since it is only able to detect outliers that have very unusual contrast values, i.e. segmentation errors. The five dimensional approach, is best suited to find most, since it is the only approach that uses all kinds of features by itself.

Combined3 gives the results for the combination of the two and three dimensional feature sets. It can be seen here, that the number of hit outliers rises significantly in comparison to the single methods, so that this combination is feasible. Recall reaches 88%, while precision is at 58%, resulting in the best F_1 value of all tested approaches. As expected, the combination of the two and the five dimensional feature set is not as good in respect to precision, since for the five dimensional feature set on its own, precision is already rather low. On the other hand, the number of detected outliers, especially shorter ones is nearly complete, as nearly all outliers have been found.

Strategy	Right	Wrong	Both
Merge	51	25	0
Merge to Seg.	51	25	0
Split and Merge	43	31	3

(a) Two features and three features.

Strategy	Right	Wrong	Both
Merge	69	31	1
Merge to Seg.	75	24	0
Split and Merge	68	41	11

(b) Two features and five features.

Tab. 4: Classification results for combined approaches.

Tables 4a and 4b show the classification results for the hit outliers for the Combined3 and Combined5 approach, respectively. Since the detection is not always exact, we declare a detected outlier as classified correctly, if it intersects with an manually marked outlier of the same type, and we declare it incorrect if it does intersect with an marked outlier of the other type.

The results show, that, without any training data, there are up to 75% right classifications with the “Merge to Segmentation” merging strategy and better results for the combination with the five features approach. The lower classification rate for Combined3 seems to be due to the fact, that in some cases, the detected silhouette around obstacles is not entirely exact. As can be seen in figure 1, at some points the silhouette is inside the sky. So technically there is a slight segmentation error, that lowers the contrast, over the obstacle. We expect, that in the full adaptive context, in a first step this would be fixed and afterwards, the real obstacle would be detected as such.

6 Conclusion

In this work, we have argued, that in cases, where an outlier detection problem and an outlier classification problem, are tackled, it might be more feasible to instead regard the problem as multiple outlier detection problems and carry out the classification implicitly by the outlier detection algorithm, that detects a given outlier. For the example of outliers in silhouettes of segmentations of mountain images, we have shown, that not only such a classification is possible, but that the usage of more than one outlier detection variant even increases the total number of detected outliers.

We believe, that the basic idea of our approach, namely the usage of separate outlier detection methods and an implicit classification based on those, is adaptable to a other outlier detection problems. Future work therefore will focus on testing such frameworks on other problems.

References

[Ab16] Abdel-Sayed, Mina; Duclos, Daniel; Faÿ, Gilles; Lacaille, Jérôme; Mougeot, Mathilde: Dictionary Comparison for Anomaly Detection on Aircraft Engine Spectrograms. In: Proc. of the MLDM 2016. 2016.

[Ah15] Ahmad, Touqeer; Bebis, George; Nicolescu, Monica; Nefian, Ara; Fong, Terry: Fusion of Edge-less and Edge-based Approaches for Horizon Line Detection. In: 6th IEEE International Conference on Information, Intelligence, Systems and Applications (IISA’15), Corfu, Greece, July 6-8, 2015. IEEE, 2015.

[Ba11a] Baboud, Lionel; Čadík, Martin; Eisemann, Elmar; Seidel, Hans-Peter: Automatic Photo-to-terrain Alignment for the Annotation of Mountain Pictures. In: Proc. of the 2011 IEEE Conference on Computer Vision and Pattern Recognition. 2011.

[BA11b] Buu, Huynh Tran Quoc; Anh, Duong Tuan: Time Series Discord Discovery Based on iSAX Symbolic Representation. In: Third International Conference on Knowledge and Systems Engineering. 2011.

[Ba12] Baatz, Georges; Saurer, Olivier; Köser, Kevin; Pollefeys, Marc: Large Scale Visual Geo-Localization of Images in Mountainous Terrain. In: Computer Vision - ECCV. 2012.

[BC83] Beckman, Richard J; Cook, R Dennis: Outlier.......... s. Technometrics, 25(2), 1983.

[BL94] Barnett, Vic; Lewis, Toby: Outliers in Statistical Data. 1994.

[BSC16] Braun, Daniel; Singhof, Michael; Conrad, Stefan: AdaMS: Adaptive Mountain Silhouette Extraction from Images. In: Proc. of the MLDM 2016. 2016.

[Ca86] Canny, John: A Computational Approach to Edge Detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI-8(6), Nov 1986.

[CBG12] Catania, Carlos A; Bromberg, Facundo; Garino, Carlos García: An Autonomous Labeling Approach to Support Vector Machines Algorithms for Network Traffic Anomaly Detection. Expert Systems with Applications, 39(2), 2012.

[Ch99] Chan, Philip K; Fan, Wei; Prodromidis, Andreas L; Stolfo, Salvatore J: Distributed Data Mining in Credit Card Fraud Detection. IEEE Intelligent Systems and Their Applications, 14(6), 1999.

[CS98] Chan, Philip K; Stolfo, Salvatore J: Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection. In: KDD. volume 98, 1998.

[Es96] Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; Xu, Xiaowei: A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. of the KDD 1996. 1996.

[FP99] Fawcett, Tom; Provost, Foster: Activity Monitoring: Noticing Interesting Changes in Behavior. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. 1999.

[Ha80] Hawkins, Douglas M: Identification of Outliers. 1980.

[KA12] Khanh, Nguyen Dang Kim; Anh, Duong Tuan: Time Series Discord Discovery Using WAT Algorithm and iSAX Representation. In: Proceedings of the Third Symposium on Information and Communication Technology. 2012.

[Ki11] Kim, Byung-Ju; Shin, Jong-Jin; Nam, Hwa-Jin; Kim, Jin-Soo: Skyline Extraction Using a Multistage Edge Filtering. World Academy of Science, Engineering and Technology, 55, 2011.

[KLF05] Keogh, Eamonn; Lin, Jessica; Fu, Ada: Hot Sax: Efficiently Finding the Most Unusual Time Series Subsequence. In: Fifth IEEE International Conference on Data Mining (ICDM'05). 2005.

[KS09] Kawahara, Yoshinobu; Sugiyama, Masashi: Change-Point Detection in Time-Series Data by Direct Density-Ratio Estimation. In: Proc. of 2009 SIAM International Conference on Data Mining (SDM2009),. 2009.

[La04] Laskov, Pavel; Schäfer, Christin; Kotenko, Igor; Müller, K-R: Intrusion Detection in Unlabeled Data with Quarter-sphere Support Vector Machines. Praxis der Informationsverarbeitung und Kommunikation, 27(4), 2004.

[PLD10] Pham, Ninh D; Le, Quang Loc; Dang, Tran Khanh: HOT aSAX: A Novel Adaptive Symbolic Representation for Time Series Discords Discovery. In: Asian Conference on Intelligent Information and Database Systems. 2010.

[SBC16] Singhof, Michael; Braun, Daniel; Conrad, Stefan: Finding Trees in Mountains – Outlier Detection on Polygonal Chains. In: Proc. of the Conference LWDA 2016. 2016.