Anomaly Detection of Trajectories with Kernel Density Estimation by Conformal Prediction

James Smith[1], Ilia Nouretdinov[1], Rachel Craddock[2], Charles Offer[2], and Alexander Gammerman[1]

[1] Computer Learning Research Center, Royal Holloway University of London
{James.Smith.2009,alex,ilia}@cs.rhul.ac.uk
[2] Thales UK
{firstname.lastname@uk.thalesgroup.com}

Abstract. This paper describes conformal prediction techniques for detecting anomalous trajectories in the maritime domain. The data used in experiments were obtained from Automatic Identification System (AIS) broadcasts – a system for tracking vessel locations. A dimensionality reduction package is used and a kernel density estimation function as a non-conformity measure has been applied to detect anomalies. We propose average p-value as an efficiency criteria for conformal anomaly detection. A comparison with a k-nearest neighbours non-conformity measure is presented and the results are discussed.

1 Introduction

Anomaly detection is a large area of research in machine learning and many interesting techniques have been developed to detect 'abnormal' behaviour of objects. The word 'anomaly' here is used in the sense that there are some patterns in the data that do not conform to typical behaviour. These non-conforming patterns are often called 'anomalies' or 'abnormalities' or 'outliers' [1]. Recently some new techniques known as conformal predictors (CP) have emerged which allow the detection of the non-conformal behaviour of objects using some measures of non-conformity [2,3]. This technique also has an advantage in delivering provably valid confidence measures under the exchangeability assumption that is usually weaker than those traditionally used.

Consider, for example, a set of moving objects (vessels, vehicles, planes, etc.) $z_1, z_2, ...$ and this movement might be normal (typical, conformal) or anomalous (atypical, non-conformal). We make an idealised assumption that $z_1, z_2, ...$ are from the same probability distribution P on the measurable feature space X independent from each other; however no further assumptions are made about P, which is completely unknown.

In this paper, the problem of anomaly detection in the maritime domain deals with trajectories of the ships to detect suspicious behaviours: a sudden change of direction, or speed, or anchoring, etc.

There has been previous research in applying conformal prediction to anomaly detection in the maritime surveillance domain [5]. Those methods focus on non-

L. Iliadis et al. (Eds.): AIAI 2014 Workshops, IFIP AICT 437, pp. 271–280, 2014.

conformity measures using nearest neighbours with Hausdorff distance or local densities of neighbourhoods with Local Outlier Factor [4,7].

In this paper we experiment with two different measures of non-conformity. In particular, the nearest neighbours non-conformity measure and the kernel density non-conformity measure have been used to detect anomalies. The data was obtained from Automatic Identification System (AIS) – a tracking system for vessels that is used to broadcast the location (retrieved by a GPS receiver onboard) of a vessel over radio-waves every few seconds.

The remaining part of this paper describes some details of conformal predictors including non-conformity measures, efficiency (performance) criteria, then a dimensionality reduction package T-SNE, the description of the data and the results with discussion. In particular, we propose average p-value as a level-independent criterion for assessing the efficiency.

2 Method

2.1 Conformal Prediction

Conformal prediction is a framework that allows making predictions with valid measures of confidence. Conformal Anomaly Detection (CAD) is an extension of Conformal Prediction that focuses on one-class (normal) in the unsupervised or semi-supervised setting [4].

Conformal Anomaly Detection

Input : Non-Conformity Measure A, significance level ϵ, training objects $z_1, z_2, ..., z_{n-1}$ and new object z_n
Output: P-value p_n, boolean variable Anomaly

$D = \{z_1, ..., z_n\}$
for $i \leftarrow 1$ **to** n **do**
  $\alpha_i \leftarrow A(D \setminus z_i, z_i)$
$\tau \leftarrow U(0,1)$
$p_n \leftarrow \frac{|\{i:a_i>a_n\}|+\tau|\{i:a_i=a_n\}|}{n}$
if $p_n < \epsilon$ **then**
  $Anomaly_j \leftarrow$ **true**
else
  $Anomaly_j \leftarrow$ **false**

Basically, the method tests whether a *new object* z_n might be generated by the same distribution as the previous *(training)* objects $z_1, \ldots, z_{n-1}$. If produced *p-value* p_n is small, then the hypothesis of the new object's normalcy is likely to be rejected, so the abnormality is confirmed.

The *significance* level ϵ regulates the pre-determined level of confidence. According to the validity property[2], if all the data objects $z_1, \ldots, z_n$ are really generated by the same distribution, then probability that $p_n \leq \epsilon$ is at most ϵ. In the context of anomaly detection this means that if z_n is not an anomaly, it

will be classified as anomaly with probability at most ϵ. This allows the false positive rate to be calibrated with a significance level parameter ϵ [7].

Another goal is efficiency: if z_n is an *anomaly*, we wish this to be captured by our test assigning a small p-value.

This performance depends on the selection of a *Non-Conformity measure (NCM)* denoted as A – that is a sort of information distance between an object and a set of the same type objects.

In this paper we use leave-one-out cross-validation to evaluate the performance. For each object a p-value is calculated using the rest of the objects as a training set. One advantage of using leave-one-out is independence on the order of data objects. Another is that it allows doing fair cross-validation using dimensionality reduction just once.

2.2 Performance Criterion

The validation of leave-one-out is done with *supervised anomaly detection* which has labelled anomalies and normal objects (from a *testing set*) where the correctness of output can be checked.

As mentioned above, the output (and the performance measure) typically depend on the significance level ϵ. Using a fixed ϵ, the more objects are classified as anomalies, the more sensitive the p-values as a test for randomness.

For supervised anomaly detection, to get an overall performance measure, independent of ϵ we adopt the well-known measure *receiver operating curve* (ROC) and use the– *area under ROC curve* (AUC) . For each value of ϵ we can produce two statistics: the percentage of normal objects classified as normal objects (that is close to $1 - \epsilon$ by validity), and the percentage of captured anomalies. AUC is the area under the corresponding ϵ-parametrized two-dimensional curve inside the square $[0, 1]^2$.

In [7], partial AUC (pAUC) is suggested for conformal anomaly detection, because it is important to minimise the number of false positives. pAUC only considers a subsection of the AUC, in particular false positive rate $\in [0, 0.01]$ and pAUC is normalised such that its outputs are $\in [0, 1]$.

Another important goal is to make the size of the *prediction set* as small as possible. The prediction set is the set of all the possible objects z_n from the feature space such that $p(z_n) \geq \epsilon$. Such kind of performance measure was investigated in the context of anomaly detection by Lei et al. [8].

We propose a new ϵ-independent version of this performance measure called *average p-value* (APV). Average p-value is the p-value of a potential new object, averaged over its location in the feature space. Its approximation can be calculated by using a finite grid of points uniformly spaced out. Every object in the grid will have a p-value calculated using a training set. The training set is fixed for each element of the grid. In the online setting it is possible to generate an APV after each iteration (using the set of normal examples as the training set). In this paper we choose to use all normal and abnormal objects in the training set to match the leave-one-out setting. We recommend using the min and max points from observed data to be used as the corners of the grid. A grid of g^d cells

is generated where g is the grid resolution, and d is number of dimensions of the feature space. A p-value is generated for each cell using the center point of each cell as the object to be evaluated. In this paper we use $g = 100$ and $d = 2$ to give a grid of 100x100 cells.

An alternative setting is the *unsupervised anomaly detection* setting which is designed for when either the data is unlabelled or no examples of anomalous objects have been provided. It considers the whole feature space as an 'ideal' testing set and considers its training set $z_1, \ldots, z_{n-1}$ as normal. In this setting AUC, and pAUC are not applicable as they both require labels, however APV could be used as a criterion in this setting. In the supervised setting AUC is preferable to APV for measuring performance, but APV can still provide information on the efficiency of non-conformity measures.

2.3 Non-conformity Measures

In this work we consider two non-conformity measures: the first is based on Kernel Density Estimation (KDE) and another, for comparison, on Nearest Neighbours algorithm. Lei et al. [8] considered KDE as a conformity measure in the unsupervised setting.

We shall start with the Kernel Density Estimation (KDE) measure. It allows assessing non-conformity based on the density of data points. The normal objects usually are concentrated in a relatively small areas (high density areas or clusters) while anomalies will be outside of these clusters. This can be exploited by estimating a probability density function from empirical data set. A standard method to do this is to use kernel density estimation. It is a non-parametric technique that requires no knowledge of the underlying distribution.

We can interpret a density function as a measure of conformity – many similar type of data points will be located together; hence we can multiply it by minus one to convert it to a non-conformity measure for consistency.

Input : Object z_i, Set of objects $z_1, z_2, ...z_n$ (note in this setup z_i is included in the set), bandwidth h, Kernel function K, number of dimensions d

Output: Non-conformity score A

$$A_i = -\left(\frac{1}{nh^d}\sum_{j=1}^{n} K\left(\frac{z_i - z_j}{h}\right)\right)$$

Kernel density estimators use the previous objects with a bandwidth parameter h that specifies the width of each object.

We will treat the bandwidth uniformly in each dimension, and fixed for each object. A kernel K is a symmetric function centred around each data point. In this work we use a Gaussian Kernel function:

$$K(u) = (2\pi)^{-d/2} e^{-\frac{1}{2}u^T u}$$

Lei et al. [8] have carried out work extending conformal prediction to produce minimal prediction regions with the use of kernel density estimators and

initially proposed KDE as a conformity measure in the unsupervised setting. Their method is underpinned by utilizing a custom bandwidth estimator that minimises the Lebesgue measure of the prediction set in the space.

We have not applied any bandwidth estimators in this paper because we wish to compare KDE with another method that also has a parameter and test performance for the parameters against multiple performance criterion.

We also apply k-Nearest Neighbour (kNN) NCM [5]: d_{ij}^{+} is the jth nearest distance to an object z_i from other objects.

Input : Object z_i, Set of objects $z_1, z_2, ...z_n$, number of nearest neighbours k

Output: Non-Conformity score A

$A_i = \sum_{j=1}^{k} d_{ij}^{+}$

The nearest neighbour non-conformity measure was found to be useful in detecting anomalies [5] and we shall use it to compare performance with the KDE NCM.

2.4 Dimensionality Reduction

The dimensionality of trajectory data is high ($4N$, where N is a number of points in a trajectory) and in order to apply kernel density estimation, we need to decrease the dimensionality of our data.

This is achieved by applying a package called T-SNE – a dimensionality reduction system. The t-Distributed Stochastic Neighbour Embedding (T-SNE) algorithm [9] is a non-deterministic and effective dimensionality reduction algorithm. It has been primarily used for visualisation but we use it to transform our data to lower-dimensional space to evaluate non-conformity measures.

In this particular application of T-SNE to trajectory data we replaced the Euclidean pairwise distance matrix with the Hausdorff distance matrix [4], but otherwise use the standard MATLAB implementation[1]. To remind the reader that the directional Hausdorff distance $\overrightarrow{H}(F,G)$ is the distance from set F to set G. $H(F,G)$ is the symmetrical Hausdorff distance. Hausdorff uses a distance metric $dist$ between the sets of points:

$$\overrightarrow{H}(F,G) = \max_{a \in F} \left\{ \min_{b \in G} \{dist(a,b)\} \right\}$$

$$H(F,G) = \max \left\{ \overrightarrow{H}(F,G), \overrightarrow{H}(G,F) \right\}$$

3 Data

An object in our task is a trajectory that can be represented as a function of position over time. We convert the trajectories into a sequence of discrete $4D$ points (x, y, x_{speed}, y_{speed}) in a similar method to [4].

[1] `http://homepage.tudelft.nl/19j49/t-SNE.html`

The original broadcasts are interpolated at a sampling distance of 200m.

If a vessel leaves the observation area for a period of 10 minutes or more, or if the vessel is stationary for a period of 5 minutes or more we consider this as the end of a trajectory. Therefore a *trajectory* is a sequence of $4D$ Points and can have any length. The $4D$ points are normalised such that $x, y \in [0, 1]$ and $x_{speed}, y_{speed} \in [-1, 1]$.

The Portsmouth dataset we evaluate was collected from a single AIS receiver on the south coast of England, during July of 2012 for one week. We filtered the data such that it only contains AIS broadcasts that report their location in a specific area between the Isle of Wight and Portsmouth. This was done to ensure consistency as the further an AIS broadcast travels the more likely it is to be lost and the data becomes less reliable.

In this dataset we consider only passenger, tanker and cargo vessels to reflect a degree of 'regular' behaviour (going from A to B and back). We assume that this data does not contain anomalous behaviour. To add anomalies we artificially inserted two sources of anomalies data points. The first contains 22 search & rescue helicopter trajectories. The other source is 180 'artificial' anomalies: random walks that have been generated starting from a random position of a random observed normal vessel. They follow a random direction and speed and a new point is generated every 200m as it has been suggested in [5]. However, unlike in [4] we only consider the entire trajectory and do not calculate detection delay.

Instead of generating anomalous trajectories of 3km in length we are using different length of "artificial" anomalies. The composition of our 180 'artificial' trajectories is the following: 150 200m long, 20 400m long, 10 600m long, 10 800m long and 10 that are 1000m long. The aim is to diversify the difficulty by providing both easy and difficult anomalies to detect.

The dataset consists of 1124 normal trajectories with 202 anomalies added to it. All these trajectories can be seen in Fig 1.

Prior to applying conformal prediction we run the T-SNE algorithm to produce $2D$ representations of the trajectories.

4 Results

For measuring the performance of the non-conformity measures we use AUC as introduced in section 2.2. The partial AUC (pAUC) is also used to show performance for $fpr \in [0, 0.01]$, note that pAUC is normalised to be in the range [0,1]. The average p-value (APV) introduced in section 2.2 is calculated, recall the lower the APV the more efficient the classifier.

AUC and pAUC are our criteria for anomaly detection ability in the supervised setting and the average p-value in the unsupervised setting which doubles as a measure of efficiency. We compare both non-conformity measures for the best parameter values of AUC, pAUC and APV. The APV, AUC and pAUC for various parameter values of both NCMs can be found in the Table 1.

Table 2 was created to expand upon the k neighbours parameter as it is apparent that the highest AUC k-NN classifier was not in the initial parameter

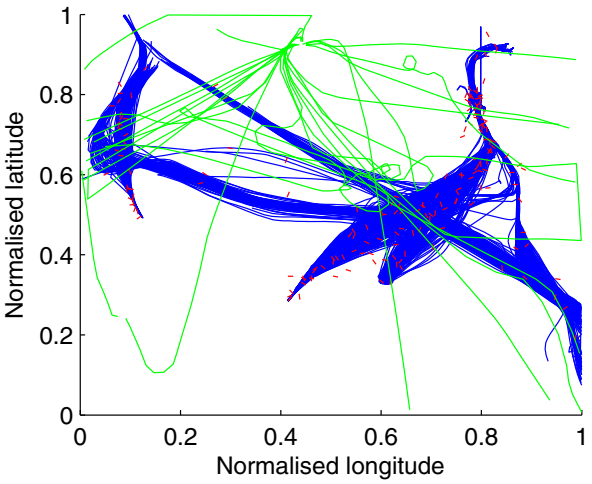

Fig. 1. Blue shows normal trajectories. Red shows the last 200m of the artificial anomalous trajectories. The green trajectories are the helicopters.

set. A rather important thing to note with testing leave-one-out is that anomalies are part of the training set, in practical applications ideally the training set would not contain anomalies. From the tables for all the parameters the highest AUC (supervised setting) are 0.7830 for KDE ($h = 3$) and 0.7616 for kNN ($k = 80$), it is clear that KDE has the higher AUC, and is therefore better at detecting anomalies across all ϵ in the leave-one-out setting. For both these parameters k-NN ($k = 80$) also has a larger APV 0.0638 against KDE ($h = 3$) 0.0606 which indicates that KDE is more efficient and offers better performance than k-NN when AUC is the criterion.

When we consider the most efficient APV (unsupervised setting) as a criterion k-NN's best parameter is $k = 7$ with APV of 0.0453 , whilst KDE's smallest APV is 0.0441 for $h = 1$.

The optimal result for the supervised problem requires more neighbours (k=80) than the unsupervised one (k=7) because most of the anomalies are close to each other (concentrating in a small area on Fig.2) which makes this problem harder. At the same time their influence on the unsupervised prediction is relatively small.

The pAUC Criterion in our leave-one-out setting may not be appropriate as the number of anomalies is far greater than a 1% composition of the dataset, but it is still a vital criterion for the purpose of minimising the false positive rate. KDE's best parameter by pAUC is $h = 2$ with a pAUC 0.484 and k-NN's best pAUC is with $k = 10$ with 0.484, however with these parameters $k = 10$ has a smaller APV and is thus more efficient. k-NN also achieves higher pAUC for more parameter values than KDE. This is quite apparent through with pAUC > 0.03 for $k = 7$ to $k = 20$, and for $k = 40$ to $k = 100$, where as for KDE only $h = \{2, 7, 8\}$ has pAUC above 0.03.

Table 1. AUC, APV and pAUC for various parameters of k-NN and KDE NCMs

k (k-NN) or h (KDE)	1	2	3	4	5	6	7	8	9	10
KDE AUC	0.6116	0.7620	**0.7830**	0.7455	0.6727	0.5932	0.5086	0.4406	0.3811	0.3518
k-NN AUC	0.2977	0.3051	0.3407	0.3611	0.3894	0.4066	0.4193	0.4323	0.4466	0.4618
KDE APV	**0.0441**	0.0519	0.0606	0.0694	0.0801	0.0936	0.1103	0.1307	0.1575	0.1941
k-NN APV	0.0490	0.0481	0.0469	0.0461	0.0458	0.0454	**0.0453**	0.0453	0.0455	0.0456
KDE pAUC	0.0082	**0.0484**	0.0285	0.0235	0.0270	0.0297	0.0415	0.0342	0.0250	0.0000
k-NN pAUC	0.0001	0.0099	0.0099	0.0099	0.0150	0.0276	0.0340	0.0381	0.0427	**0.0484**

Table 2. Extension of k-NN results

k (k-NN)	20	30	40	50	60	70	80	90	100	110
K-NN AUC	0.6157	0.7051	0.7257	0.7403	0.7519	0.7547	**0.7616**	0.6832	0.6301	0.6032
k-NN APV	0.0486	0.0519	0.0549	0.0574	0.0595	0.0615	0.0638	0.0705	0.0827	0.0954
k-NN pAUC	0.0304	0.0253	0.0327	0.0308	0.0400	0.0381	0.0384	0.0434	0.0375	0.0110

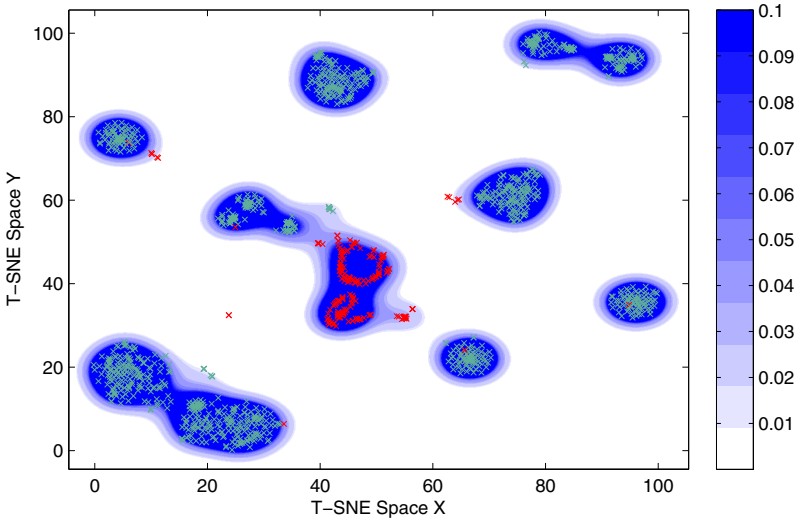

Fig. 2. Prediction sets for various parameters of ϵ in T-SNE space for KDE NCM (h=3). The labelled colours are for various values of ϵ, the red objects are anomalies, and the teal colour is used for the normal trajectories.

Figure 2 visualises the 'normal' class prediction sets of various ϵ for the KDE NCM. It is generated using a grid as the test set and all the objects from our dataset as the training set for each object in the grid.

5 Conclusions and Future Work

In this paper we applied conformal anomaly detection and studied applicable performance measures of efficiency. We have seen that it may be problematic to evaluate the performance of conformal anomaly detection directly because usually either the amount of labelled data for testing the accuracy is small or these data are not representative enough. Therefore we propose average p-value. Average p-value is an as a performance criterion that works in both the supervised and unsupervised settings and does not require labelled anomalies to evaluate performance. At the same time, it is independent on the significance level.

However, we applied some supervised criteria as well.

As examples of NCMs, we used two methods based on the idea of density approximation. One of them is nearest neighbours (k-NN) algorithm and the other is kernel density estimation (KDE) that considers an entire trajectory to the maritime surveillance domain. In addition, we reduced the dimensionality of our dataset to compare the different non-conformity measures.

In the leave-one-out supervised setting KDE NCM for our dataset in the supervised leave-one-out setting has higher AUC than the k-NN NCM. However for most anomaly detection applications performance at small false positive rates is more important. If small false positive rate (in the form of pAUC) is the primary criterion then k-NN NCM performs better than the KDE NCM.

Going to average p-value we see that KDE can lead to more efficient predictions with a smaller average p-value than k-NN, this indicates KDE NCM in the unsupervised setting with a good choice of parameter performs better with our dataset than the k-NN NCM.

Both KDE NCM and k-NN NCM performances for all criterion are dependent on the choice of parameter h and k respectively. We included some observations related to that.

For future work, it would be interesting to continue the work using other sources of data and to reach some explanation of the noticed effects. We also plan to apply various other NCMs in search of anomalous objects.

Acknowledgements. James Smith is very grateful for a PhD studentship jointly funded by Thales UK and Royal Holloway, University of London. This work is supported by EPSRC grant EP/K033344/1 ("Mining the Network Behaviour of Bots"); by the National Natural Science Foundation of China (No.61128003) grant; and by grant 'Development of New Venn Prediction Methods for Osteoporosis Risk Assessment' from the Cyprus Research Promotion Foundation. We are also grateful to Rikard Laxhammar, Vladimir Vovk, Christopher Watkins and Jiaxin Kou for useful discussions. AIS Data was provided by Thales UK.

References

1. Chandola, V., Banerjee, A., Kumar, V.: Anomaly Detection A Survey. ACM Computing Surveys (CSUR) (2009), `dl.acm.org`
2. Vovk, V., Gammerman, A., Shafer, G.: Algorithmic learning in a random world. Springer (2005)

3. Gammerman, A., Vovk, V.: Hedging predictions in machine learning. The Computer Journal 50(2), 151–163
4. Laxhammar, R., Falkman, G.: Sequential Conformal Anomaly Detection in Trajectories based on Hausdorff Distance. In: 2011 Proceedings of the 14th International Conference on Information Fusion (FUSION) (2011)
5. Laxhammar, R., Falkman, G.: Conformal prediction for distribution-independent anomaly detection in streaming vessel data. In: Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques, pp. 47–55. ACM (2010)
6. Laxhammar, R., Falkman, G.: Online Detection of Anomalous Sub-trajectories: A Sliding Window Approach Based on Conformal Anomaly Detection and Local Outlier Factor. In: Iliadis, L., Maglogiannis, I., Papadopoulos, H., Karatzas, K., Sioutas, S. (eds.) AIAI 2012, Part II. IFIP AICT, vol. 382, pp. 192–202. Springer, Heidelberg (2012)
7. Laxhammar, R.: Conformal anomaly detection: Detecting abnormal trajectories in surveillance applications. PhD Thesis, University of Skovde (2014)
8. Lei, J., Robins, J., Wasserman, L.: Distribution-Free Prediction Sets. Journal of the American Statistical Association 108(501), 278–287 (2013)
9. Van der Maaten, L., Hinton, G.: Visualizing Data using t-SNE. Journal of Machine Learning Research 9(11) (2008),
`http://homepage.tudelft.nl/19j49/t-SNE.html`