Clustering-Based Anomaly Detection in Multi-View Data

Alejandro Marcos Alvarez*
University of Liège
amarcos@ulg.ac.be

Makoto Yamada
Yahoo! Labs
makotoy@yahoo-inc.com

Akisato Kimura
NTT CS Labs
akisato@ieee.org

Tomoharu Iwata
NTT CS Labs
iwata.tomoharu@lab.ntt.co.jp

ABSTRACT

This paper proposes a simple yet effective anomaly detection method for multi-view data. The proposed approach detects anomalies by comparing the neighborhoods in different views. Specifically, clustering is performed separately in the different views and affinity vectors are derived for each object from the clustering results. Then, the anomalies are detected by comparing affinity vectors in the multiple views. An advantage of the proposed method over existing methods is that the tuning parameters can be determined effectively from the given data. Through experiments on synthetic and benchmark datasets, we show that the proposed method outperforms existing methods.

Categories and Subject Descriptors

H.3.3 [**Information systems**]: Information Storage and Retrieval—*Information Search and Retrieval*

Keywords

Anomaly detection; multi-view data; affinity propagation

1. INTRODUCTION

Anomaly detection methods aim at identifying anomalies, a.k.a. outliers, in a given dataset. Those methods are useful for various types of applications including fraud and intrusion detection [2]. The standard anomaly detection methods use single-view data to identify anomalies. For example, the one-class support vector machine approach [7] finds a *normal* region containing a certain fraction of samples and regards the samples not included in the region as anomalies.

However, present-day data tends to be multi-view, i.e. the individual objects are described from several perspectives or several views, each of which highlights different characteristics of the objects. An example of multi-view data is the data acquired by stereo cameras that capture the images from two different points of view. Another appealing example appears in social networks where multimedia content (texts, images, movies, etc.) can be described by its intrinsic information (bag of words, image features, etc.) or by the actions of the users (comments, 'like', number of views, etc.). In order to adapt to this particular property, anomaly detection methods should be designed to take advantage of those multiple views.

Recently, Gao et al. [4] proposed a novel multi-view anomaly detection algorithm, called *horizontal anomaly detection* (HOAD), that exploits several different data sources, i.e. multi-view data, to detect outliers of the dataset. The main idea behind HOAD is to regard samples whose behavior is *inconsistent* among multiple information sources as anomalies. More specifically, HOAD first clusters objects simultaneously in all the views with spectral clustering [6] and then classifies as anomalies the samples that belong to different clusters in different views. Gao et. al experimentally showed the efficiency of their approach. However, HOAD has several tuning parameters, including the number of clusters, which is assumed to be the same for all views. As a consequence, the performance of HOAD highly depends on the values given to the tuning parameters.

To overcome the drawbacks of HOAD, we propose a multi-view anomaly detection method based on affinity propagation (AP) [3]. More specifically, we first cluster the data independently in each view with AP and then compute, from the clustering results, affinity vectors whose elements represent the affinity between two objects. Finally, we detect anomalies by simply comparing anomaly scores computed from the affinity vectors. An advantage of the proposed method over HOAD is that the tuning parameters can be easily determined by a simple, yet effective, heuristic. In addition, the number of clusters is automatically determined by AP, while HOAD needs to be given the number of clusters beforehand. This is an important property for the practitioner. Through experiments on synthetic and benchmark datasets, the proposed method is shown to outperform HOAD.

2. PROBLEM FORMULATION

In this section, we give a formal definition of the multi-view anomaly detection problem.

*This work was completed when the author was intern at NTT CS Labs.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2505515.2507840.

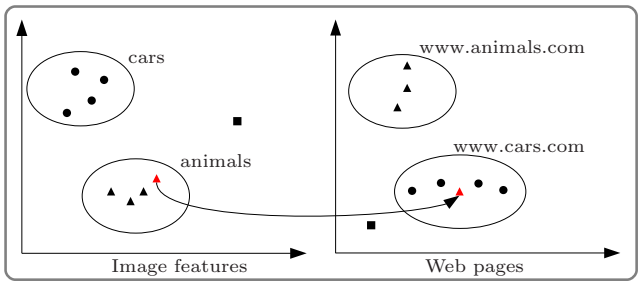


Figure 1: Illustrative example: motivation of using multi-view data in anomaly detection: with only one view, some anomalies may remain hidden.

Suppose we are given n independent and identically distributed (i.i.d.) objects, denoted by $i = 1, \ldots, n$, described by V views denoted by $v = 1, \ldots, V$. Each object i, when seen through view v, has a feature representation $\boldsymbol{x}_i^v$ such that:

$$\boldsymbol{x}_i^v \in \mathcal{X}^v \subset \mathbb{R}^{d^v},$$

where $\mathcal{X}^v$ is the domain of view v and d^v is the dimensionality of that domain. In this case, $\boldsymbol{x}_i^v$ is the feature representation of object i when seen from the particular perspective of view v. Note that each view sees different aspects of the object. Consequently, the object has V different feature representations, one for each view.

We define D as the set containing the V feature representations of each object i:

$$D = \left\{ \left(\boldsymbol{x}_i^1, \ldots, \boldsymbol{x}_i^V \right) \right\}_{i=1}^{n}.$$

Then, our multi-view anomaly detection strategy consists in computing an anomaly score for each object and in comparing this anomaly score to a threshold τ that controls the sensibility/robustness tradeoff.

3. PROPOSED METHOD

In this section, we propose a method that detects anomalies in multi-view data.

3.1 Anomaly Detection using Affinity Information

In this paper, we propose a new multi-view anomaly detection method that consists in three steps: (1) perform clustering separately in each view, (2) for each view, create an affinity vector for each object based on the clustering results of step (1), and (3) compute an anomaly score based on affinity vectors. The key idea here is to utilize clustering-based affinity information to detect anomalies. The proposed method is designed to detect the anomalous objects whose neighborhoods are inconsistent among multiple views.

Figure 1 illustrates our multi-view anomaly detection framework using an example with web images observed from two points of view: the image features and the web pages in which the images are contained. The leftmost graph shows the objects (clustered in 'animals' and 'cars') from the image features view, while the rightmost graph shows the objects (clustered in 'animals.com' and 'cars.com') from the web pages view.

If we consider only a single view, i.e. image features or web pages, then, only the image represented by a square can be identified as a clear anomaly while the rest of the data seems normal. However, if we consider the two views simultaneously, the square is not an outlier anymore because its neighborhood is consistent in both views. In this case, the square is far away from the other objects in both views. On the other hand, when both views are considered, another point draws our attention. Indeed, we notice that the red triangle is clustered in 'animals' in the image features view, while it is clustered in 'cars.com' in the web pages view. The clustering-based affinity information, or neighborhoods, in the image features view and in the web pages view are *inconsistent*. We define this behavior as anomalous and regard the red triangle as an anomalous image.

3.2 Affinity Propagation

In this paper, we employ *affinity propagation* (AP) [3] as a clustering method. AP is an exemplar based clustering algorithm that associates each object with its exemplar. In this context, an exemplar is an object that corresponds to the center of one cluster. We denote that object i is associated with exemplar j by $c_i = j$. In this setting, all the configurations are not allowed and some constraints are needed to ensure that the result is coherent. Indeed, if object i is associated with exemplar j ($c_i = j$), then the exemplar of j should be j as well ($c_j = j$), if not, this means that object j is actually not an exemplar.

Affinity propagation computes two messages for each pair (i, j) of objects: the responsibility r_{ij} and the availability a_{ij}. The responsibility r_{ij} sent from i to j models how much object i wants object j as exemplar, whereas the availability a_{ij} sent from j to i models how much object j wants to be the exemplar of object i. As AP is an iterative algorithm, the responsibilities and availabilities are updated at each iteration until convergence is reached or until a fixed budget of iterations is exhausted. The initial availabilities are set to 1 and the update equations are as follows:

$$a_{jj} = -1 + \prod_{k:k\neq j} (1 + r_{kj}),$$

$$a_{ij} = \left[1 + \left(\frac{1}{r_{ii}} - 1 \right) \prod_{k:k\neq i, k\neq j} (1 + r_{ki})^{-1} \right]^{-1},$$

$$r_{ij} = \boldsymbol{L}_{ij} \left[\sum_{k:k\neq j} a_{ik} \boldsymbol{L}_{ik} \right]^{-1},$$

where $\boldsymbol{L}$ is the likelihood matrix such that $\boldsymbol{L}_{ij}$ represents the affinity, or similarity, between objects i and j. A large value of $\boldsymbol{L}_{ij}$ indicates that the objects are similar. Note that, for AP, we need to set the diagonal values of $\boldsymbol{L}$. Smaller values on the diagonal tend to yield a smaller number of clusters. Frey et al. [3] propose to set those diagonal values to either the median or the minimum of the non-diagonal elements of $\boldsymbol{L}$. Those heuristics work very well in practice.

Once the algorithm has converged, the clustering solution is obtained $\forall i$ by $c_i = \{1, \ldots, n\}$, such that

$$c_i = \arg\max_j \left[a_{ij} + r_{ij} \right],$$

where the value of c_i represents the exemplar of object i.

In the proposed method, we apply affinity propagation separately in the different views. Consequently, we have V clustering results that we refer to as

$$\left\{ \left\{ c_i^v \right\}_{i=1}^n \right\}_{v=1}^V,$$

such that the clustering results obtained in view v are denoted by c_i^v.

3.3 Clustering-Based Affinity Vectors

Now, we define the clustering-based affinity vector $\boldsymbol{z}_i^v \in \mathbb{R}^n$ of object i in view v. We propose to compute $\boldsymbol{z}_i^v$ as follows:

$$\boldsymbol{z}_i^v(j) = \begin{cases} 0 & \text{if } i = j, \\ Z^{-1} \exp\left(\boldsymbol{L}_{ic_j^v}^v + \boldsymbol{L}_{ij}^v - 2 \right) & \text{otherwise,} \end{cases}$$

where $\boldsymbol{z}_i^v(j)$ is the j-th element of vector $\boldsymbol{z}_i^v$, Z is a normalization factor such that $\sum_j \boldsymbol{z}_i^v(j) = 1$, and $\boldsymbol{L}_{ic_j^v}^v$ is the likelihood between object i and the exemplar of j, c_j^v, in the view v. The key idea here is to incorporate clustering information into $\boldsymbol{z}_i^v$.

If two objects i and j are similar, we expect their likelihood $\boldsymbol{L}_{ij}^v$ to be large and their exemplar to be the same, thus giving a large value of $\boldsymbol{z}_i^v(j)$. On the other hand, if two objects are assigned to different clusters or if their likelihood $\boldsymbol{L}_{ij}^v$ is rather small, then the value $\boldsymbol{z}_i^v(j)$ is small. Thus, this definition of $\boldsymbol{z}_i^v$ characterizes the affinity of the objects with respect to object i and to the current clustering structure. Indeed, when $i \neq j$, $\boldsymbol{z}_i^v(j)$ can be regarded as a similarity score between object i and object j with respect to the clustering structure in the v-th view.

3.4 Anomaly Score Computation

We propose several types of anomaly score computation. Here, we consider the two-view case. However, the proposed method is much more general and can be adapted without effort to the case where more than two views are considered.

In any case, object i is flagged as an anomaly if the score $A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right)$ is smaller than the threshold τ.

Distance based: We define a first anomaly score as:

$$A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right) = \frac{1}{\|\boldsymbol{z}_i^1 - \boldsymbol{z}_i^2\|^2}.$$

Pearson's correlation based: $A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right)$ can also be computed from the Pearson's correlation between the affinity vectors $\boldsymbol{z}_i^v$ of object i in both views.

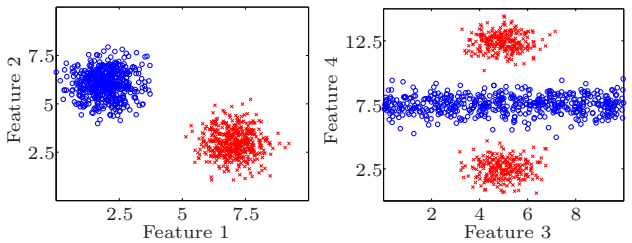
Spearman's correlation based: We define another anomaly score $A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right)$ as being the Spearman's correlation between the affinity vectors $\boldsymbol{z}_i^v$ of object i in both views.

Independence based: The last anomaly score is given by

$$A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right) = \Delta_{ii},$$

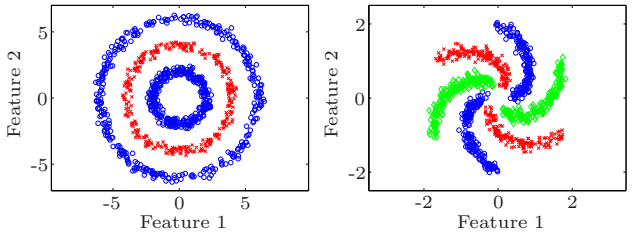
with

$$\Delta = \boldsymbol{H}\left(\boldsymbol{Z}_1 + \boldsymbol{Z}_1^\top\right)\boldsymbol{H}\left(\boldsymbol{Z}_2 + \boldsymbol{Z}_2^\top\right),$$
$$\boldsymbol{H} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top,$$
$$\boldsymbol{Z}_v = \left[\begin{array}{cccc} \boldsymbol{z}_1^v & \boldsymbol{z}_2^v & \ldots & \boldsymbol{z}_n^v \end{array} \right],$$



(a) Features 1 & 2 (b) Features 3 & 4

Figure 2: Synthetic dataset: 'structures' (2 classes, 2000 objects, 4 dimensions).



(a) 'circle' (b) 'spiral'

Figure 3: 2D Synthetic datasets: 'circle' (2 classes, 2000 objects) and 'spiral' (3 classes, 5000 objects).

where $\mathbf{I}_n$ is the identity matrix of dimension n and $\mathbf{1}_n$ is a n-dimensional column vector with each element equal to 1. Note that the summation of all the anomaly scores $A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right)$ can be interpreted as a variant of a kernel-based independence measure called the (empirical) *Hilbert-Schmidt Independence Criterion* (HSIC) [5]:

$$HSIC = \sum_{i=1}^n A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right) = \operatorname{tr}(\Delta),$$

where $\operatorname{tr}(\cdot)$ denotes the trace. HSIC always takes a non-negative value, and is zero if and only if two random variables are statistically independent[1], i.e. $p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x})p(\boldsymbol{y})$. Thus, a small $A\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right)$ can be interpreted as a clue that $\boldsymbol{z}_i^1$ and $\boldsymbol{z}_i^2$, i.e. the object in view 1 and view 2, are independent.

4. EXPERIMENTS

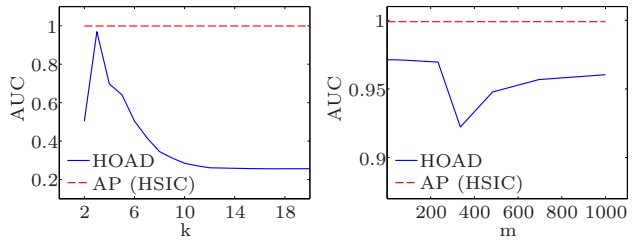
We now experimentally show the effectiveness of the proposed method on synthetic and benchmark datasets.

Datasets: We consider seven datasets for our experiments: one 4D and two 2D synthetic datasets, namely 'structures', 'circle' and 'spiral' shown in Figures 2 and 3 respectively, and four benchmark datasets, namely 'iris', 'letter', 'waveform' and 'zoo', selected by Gao et al. [4] from the UCI machine learning repository [1]. Those datasets are not multi-view datasets. We thus simulate two views by following the approach in [4]. The approach consists in splitting the object feature representation into two subsets, where each subset is considered as one view of the data. In order to generate an anomaly, we take two objects from two different classes and swap the subsets in one view but not in the other. We randomly perturb 10% of the data in that way.

[1]In theory, the independence is assured if we use a *universal reproducing kernel* such as the Gaussian kernel.

	HOAD		AP (Distance)		AP (Pearson)		AP (Spearman)		AP (HSIC)	
	L2	Gauss	L2	Gauss	L2	Gauss	L2	Gauss	L2	Gauss
structures	0.6016	0.9694	0.6248	0.9985	0.9998	0.9995	0.8789	0.8746	0.9997	**0.9999**
circle	0.5377	0.5123	0.5326	0.4999	0.7796	0.5598	0.5217	0.5207	**0.7791**	0.5822
spiral	0.5195	0.5265	0.4779	0.5218	0.6026	0.6259	0.5877	0.5879	0.6166	**0.6394**
iris	0.6520	0.8386	0.8908	0.9077	0.9585	0.9445	0.9292	0.9292	**0.9587**	**0.9508**
letter	0.5471	0.5404	0.5979	0.7439	0.8594	0.8644	0.7719	0.7731	**0.8839**	0.8717
waveform	0.7314	0.7464	0.5072	0.9154	0.6080	0.9019	0.8891	0.8889	0.6188	**0.8816**
zoo	0.5039	0.7501	0.7881	0.9446	0.9713	0.9659	0.8887	0.8845	**0.9793**	**0.9669**

Table 1: AUC values averaged over 50 runs. A statistical test is used to compare HOAD to AP (HSIC). A bold font indicates which method is significantly better according to a one-tailed Student's t-test ($p = 0.01$).



(a) Influence of k ($m^* = 23.2079$)

(b) Influence of m ($k^* = 3$)

Figure 4: Evolution of AUC with the number of clusters k and the penalty term m for HOAD method. Those results are obtained for the 'structures' dataset and are averaged over 50 runs. The Gaussian kernel is used to compute the affinity matrix.

Setup: We compare our affinity propagation based method, or 'AP', to the spectral clustering based method, or HOAD, proposed in [4]. The methods are compared based on the *area under ROC curve* (AUC) as proposed in [4]. For AP, we set the values $\boldsymbol{L}^v_{ii}$ to the median of the non-diagonal elements of $\boldsymbol{L}^v$, as proposed in [3]. As for HOAD, we need to set two parameters: k and m. Parameter k specifies the number of clusters in which HOAD should cluster the objects and m is a penalty term penalizing the clustering if one object is clustered differently in several views. For each dataset, we select, from a list of possible values, the pair of values (k^*, m^*) that give the best results on that dataset. Both AP and HOAD need a likelihood, or affinity, matrix as input. We compare two types of affinity measures. The first one is the negative L2-norm:

$$-\|\boldsymbol{x}^v_i - \boldsymbol{x}^v_j\|^2,$$

and the second one is the Gaussian kernel:

$$\exp\left(-\frac{\|\boldsymbol{x}^v_i - \boldsymbol{x}^v_j\|^2}{2\sigma^2}\right),$$

where σ is a kernel parameter whose value is given by $\sigma = 2^{-0.5}\text{median}(\{\|\boldsymbol{x}^v_i - \boldsymbol{x}^v_j\|\}^n_{i,j=1})$, following a common heuristic [8]. Each method is applied 50 times on randomly perturbed versions of each dataset by using the aforementioned trick to generate the anomalies.

Results: Figure 4 first shows the influence of parameters k and m on the performance of HOAD on the synthetic dataset 'structures'. The performance critically depends on the values given to k and m. Note that, when we explore a range of possible values of parameter k, parameter m is fixed to its optimal value m^*. We proceed in the same way when we explore the range of possible values of m. Table 1 shows the AUC for the different settings and the different datasets. We statistically compare HOAD to AP (HSIC). A bold font indicates that the method is significantly better than the other one according to a one-tailed Student's t-test ($p = 0.01$). From these results, we clearly see that our method outperforms HOAD in all the considered datasets.

5. CONCLUSION

In this paper, we developed a new anomaly detection approach for multi-view data. The main idea behind the approach is to compare the objects based on their neighborhoods in the different views. In order to achieve that, we first perform clustering separately in the different views, and then compute clustering-based affinity vectors for each object. Each affinity vector is computed such that it takes into account the clustering results in the current view. An anomaly score is then computed for each object by comparing its affinity vectors with each other. Through experiments on several datasets, we showed that the proposed method outperforms existing methods. Moreover, the proposed method does not require a careful tuning of parameters and can automatically adjust to different clustering structures with no outside intervention.

References

[1] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[2] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 2009.

[3] B. Frey and D. Dueck. Mixture modeling by affinity propagation. In *NIPS*, 2006.

[4] J. Gao, W. Fan, D. Turaga, S. Parthasarathy, and J. Han. A spectral framework for detecting inconsistency across multi-source object relationships. In *ICDM*, 2011.

[5] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *ALT*, 2005.

[6] A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2002.

[7] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 2001.

[8] B. Schölkopf and A. J. Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press, 2001.