APPLYING ONE CLASS CLASSIFIER TECHNIQUES TO REDUCE MAINTENANCE COSTS OF EAI

Iñaki Fernández de Viana, Pedro J. Abad, José L. Álvarez and José L. Arjona
Departamento de Tecnologías de le Información, Universidad de Huelva
Escuela Técnica Superior de Ingeniería La Rábida, Huelva, Spain

Keywords: Outlier detection, One class classification, Novelty recognition, Web wrapper, Enterprise application integration.

Abstract: Reducing maintenance costs of Enterprise Application Integration (EAI) solutions becomes a challenge when you are trying to integrate friendly web applications. This problem can be solved if we use automated systems which allow navigating, extracting, structuring and verifying relevant information. The verification task aims to check if the information is correct. In this work we intend to solve the verify problem regarding One Class Classification Problem. One Class Classification Problems are classification problems where the training set contains classes that have either no instances at all or very few. During training, in the verify problem, we only have instances of the classes we know. Therefore, the One Class Classifier techniques could be applied. In order to evaluate the performance of these methods we use different databases proposed in the current literature. Statistical analyses of the results obtained by some basic One Class Classification techniques will be described.

1 INTRODUCTION

Internet is the main source of information and has been designed to be used by humans. This is a disadvantage if we want to process automatically the information contained in it. It is unusual that web sites provide a programmatic interface (API) to obtain a structured view of the relevant information they offer. As a result, the costs of integrating Enterprise Applications (EA) that use this kind of information sources are very high because they have to process unstructured data. According to a report by IBM, for each dollar spent on the development of an application, the cost to integrate it is five to twenty times higher (Weiss, 2005). Another fact is that information integration exhausts about 40% of the budget spent on information technology (Bernstein and Haas, 2008). Therefore, to reduce these costs, engineering solutions to the integration problem are a must.

In order to address this problem we use wrappers. A wrapper is a piece of software that allows a deep-web information source to be added by a virtual schemata (Madhavan et al., 2007).

In this work, we focus on verifying the data obtained by a wrapper. If a wapper lacked this element, the different applications we want to integrate could be fed with inconsistent information. Subsequently, these data could be used, for example, by enterprise decision systems which could trigger unpredictable consequences.

Information extractors, one of the steps of a wrapper, are composed of a set of extraction rules inferred from a training set. When extraction rules rely on HTML landmarks, Information Extractors can only extract information from the same information source where the training set was obtained. Therefore, if the source changes, in some cases the returned data could be incorrect (Kushmerick, 2000). Unless the information generated by wrappers is verified in an automatic way, these data can go unnoticed in applications that use them.

In general terms, the verification process begins by invoking the wrapper in order to obtain the set of results that we shall use to produce the training set. This training set is characterised by a set of numerical and categorical features. These features will be profiled and combined to model the training set. When the verifier receives an unverified result set, it will calculate the values of the features and then it will check if they fit the previously calculated model.

Fernández de Viana I., J. Abad P., L. Álvarez J. and L. Arjona J..
APPLYING ONE CLASS CLASSIFIER TECHNIQUES TO REDUCE MAINTENANCE COSTS OF EAI.
DOI: 10.5220/0003503800410046
In *Proceedings of the 6th International Conference on Software and Database Technologies* (ICSOFT-2011), pages 41-46
ISBN: 978-989-8425-76-8

As we will see, one of the drawbacks we must face during the building of the verifier is that only positive examples exist. In this work we propose solving the verify problem with the One Class Classifier Problem (OCP) viewpoint. Such a viewpoint is different from current proposals. OCP is an unsupervised classification problem where the training set contains classes that have no instances at all, very few of them, or they are not statistically representative. OCP is an important task in machine learning and data mining activities. A lot of these techniques gave good results in applications like (Hodge and Austin, 2004; Chandola et al., 2009): continuous typist recognition; fault diagnosis; detecting mislabelled data in a training data set; or detecting unexpected entries in databases.

In order to evaluate the performance of these methods we use different databases proposed in the current literature (Kushmerick, 2000; Lerman et al., 2003; McCann et al., 2005). Statistical analyses of the results obtained by some basic OCC techniques will be described. These analyses rely on the non-parametric testing techniques proposed in (García et al., 2010) to ascertain the statistical significance among the measure means of the different algorithms.

This paper is organised as follows. Section 2 explains the concepts of OCP and One Class Classifier (OCC) methods and Section 3 justifies the use of such techniques in the verify problem. We describe the set up of our experiments and discuss the results obtained by the classifiers tested in Section 4. Finally, Section 5 concludes by stating that OCC methods are competitive in performance and deserve to be applied to the verify problem as an alternative to current proposals.

2 ONE CLASS CLASSIFICATION

In most of the classification problems, the training set is composed of instances of all known classes that can appear during testing. In theses cases, conventional multi-class classification algorithms, which aim to classify unknown instances into one of the known class, can be used to determine decision boundaries that distinguish these classes. However, in other classification problem suites, the training set contains classes that have no instances at all, very few of them, or they are not statistically representative; although it is well-known that models that build on valid data only tend to overgeneralise.

For this set of problems, One Class Classifier (OCC) techniques are used (Tax, 2001). These algorithms label all instances as target if those instances belong to the learning classes in training, otherwise as outlier if those instances do not belong to these classes. The main differences between multi-class and OCC are (Tax, 2001): (i) OCC techniques only use target class information and (ii) the goal of OCC techniques is to define a boundary around the target class that accepts as many target objects as possible and minimises the chance of accepting outliers.

OCC techniques are used in environments where outliers may indicate a malfunction. Some of these applications are (Hodge and Austin, 2004; Chandola et al., 2009): continuous typist recognition; fault diagnosis; detecting mislabelled data in a training data set; or detecting unexpected entries in databases.

Due to the large number of OCC methods, in (Tax, 2001), a classification of the latter was proposed. This taxonomy groups OCC techniques into three main approaches: **Density estimation** (examples are a Gaussian model, a mixture of Gaussian and Parzen density estimators), **Boundary methods** (nethods like k-centers, NN-d and SVDD belong to this category), **Reconstruction methods** (examples of these techniques are k-mean clustering, self-organising maps, PCA, a mix of PCA and diabolo networks).

Different research (Hempstalk et al., 2008) suggest that one-class problems should be reformulated into two-class problems because the target class could be characterised as using an artificial data generator that comes from a known reference distribution such as a multi-variate normal distribution, which can be estimated from the training data for the target class. The drawback of these algorithms is that generating good negative examples is a difficult and expensive task in high dimensional spaces.

3 THE VERIFY PROBLEM AS OCP

If the verify problem is re-phrased in terms of feature vectors and how close they are to one another, the problem is then closely related to a classification problem.

The verify problem cannot be considered a multi-class classification problem because, the training set must originate from the same wrapper so that all the result sets returned by a reaper are expected to be valid. This is because if an Information Extractor collects invalid result sets and they are detected during the verifier design, the Information Extractor will be modified to avoid these mistakes.

To deal with this problem, it is necessary to create new synthetic result sets using so-called perturbations. In (McCann et al., 2005) we have found different proposals the problem with these perturbations is

that they may lead to result sets that deviate largely from current result sets.

This proposed solution for the verify field is similar to some proposals found in the one-class algorithm field, hence we can say that this one and the rest of the applied techniques in OCC fields can also be applied to the verify problem.

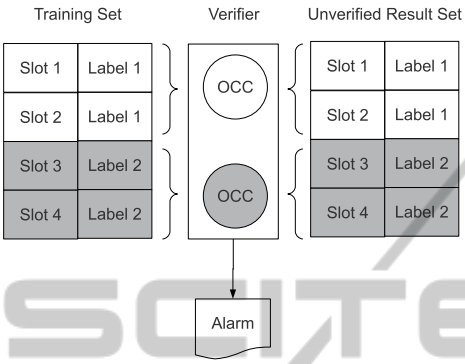

Figure 1: The verify problem as OCP.

The verify problem can be solved under OCP point of view. From each working set obtained during the reaping phase, we have a set of result sets composed of valid data. We assume that these result sets are multi-slot and every slot is labelled with a class. Besides, slots are characterised using a set of features. Our training set is the union of all result sets.

From each multi-class training set composed of *N* classes, *N* OCC methods have to be inferred. Every verification model, OCC method, is constructed using only one class (target class) instances. That is, the training set of the first OCC is composed of first class instances, the second OCC training set is composed of second class instances, and so forth.

Finally, when an unverified result set has to be verified, we must use the *N* OCC methods learned. All slots labelled as first class will be tested by the first OCC, slots labelled as second class by the second, etc. If one of them predicts that at least one slot is an outlier, an alarm will be signalled, otherwise, the result set will be labelled as correct (Figure 1).

OCP is an important unsupervised classification task in machine learning and data mining activities. A lot of these techniques gave good results in different applications.

4 EXPERIMENTS AND RESULTS

This section describes the set-up of our experiments and compare the performance of some OCC methods.

4.1 Database Description

In order to evaluate the performance of the different OCC, we used the labelled datasets proposed in (Kushmerick, 2000). This database is composed of 27 Internet sites. These sites were chosen to represent the sort of search services found on the Internet in 1998. In total, an average of 867 pages were gathered from each site, for a total of 23,416 pages. Every site has a different number of classes (labels) that ranges from 2 to 8. For our experiments, we consider that each class contained in each site is a different OCP. Hence, we will have a total of 102 OCC to train and test. The are other databases used in Information Extractor verify problems (McCann et al., 2005; Lerman et al., 2003) but they are incomplete.

Every instance must be characterised by a set of features. In our experiments 25 of the numeric features reported in (Kushmerick, 2000; McCann et al., 2005; Lerman et al., 2003; Chidlovskii et al., 2006) have been used. We have grouped them into several categories like counting attributes of a given class that match a given starting or ending pattern or counting attributes that begin with a lower-case letter.

Also, 14 of the categorical features explained in (McCann et al., 2005; Chidlovskii et al., 2006) were employed. Examples of these features are: URL attribute (returns true if the attribute is a valid URL) and Time attribute (returns true if the attribute is a valid date).

Besides, for each site 5 different experiments formed by 300 random instances of each class were carried out.

Note that a OCP is of high dimension and so it is important to use dimension reduction techniques to obtain simple models that describe our problem (Villalba and Cunningham, 2007). Techniques applied in multi-class classifiers appear not to be applicable to OCC models, because if we only have target instances, it would be difficult to identify a set of features that distinguishes them. In our experiment we use the Laplacian score for feature selection, a technique based on local preservation evaluated in (Villalba and Cunningham, 2007). Laplacian Score returns features ranked, and so we only select those that belong to the first, second or third quartile. For almost all sites the dimensionality mean decreases to 47%.

4.2 Methods used for Comparison Purposes

To evaluate the performance of OCC applied to information extractor verifiers, the dd_tools Matlab toolbox (Tax, 2009) was applied. The algorithms used in

our experiment were: gauss_dd (fits a Gaussian density on the dataset), knndd (calculates the K-Nearest neighbour data description on the dataset), parzen_dd (fits a Parzen density on the dataset), svdd (optimises a support vector data description for the dataset by quadratic programming), som_dd (self-Organising Map data description on the database).

We have chosen these methods because they represent each OCC technique sets summarised in 2.

4.3 Performance Measures and Algorithm Parameters

The OCC studied in this work were evaluated with 5 different measure: Precision (P), Recall (R), F measure (F), Accuracy (A) and Area Under Roc Curve (AUC).

We always used the mean value of these measures when we compared every OCC method evaluated. The performance of each algorithm (one site and class OCC) was evaluated using 10-fold cross-validation procedures. From these 10 sub-samples, a single sub-sample is retained as data validation to test the model and it has target and outlier class instances. The remaining 9 sub-samples are used as training data and only have target class instances.

One difficulty in assessing the performance of an OCC is the parameter tuning required for the different techniques. Because this work is a concept test regarding the use of OCC in Information Extractor verifiers, we follow a simple approach of fixing parameter values, the default values proposed in (Tax, 2009) were used. That is, each method is not trained with the parameter value that yields the best result. We only changed the percentage of targets rejected during training to 0 (overgeneralise).

4.4 Comparatives

First of all, the Kolmogorov-Smirnov test is applied to check if a normal distribution of the measured values can be taken at a 0.05 level of confidence.

If its null hypothesis is rejected, the initial conditions that guarantee the reliability of the parametric tests may not be satisfied causing the statistical analysis to lose credibility with these tests. Hence, the non-parametric testing techniques proposed in (García et al., 2010) to ascertain the statistical significance among the measures means will be considered.

Then, we compare every feature selection OCC method with itself without feature selection. To perform this comparative we use Wilconxon Signed-Ranks Test (Demsar, 2006), setting the level of confidence at 0.05.

After studying if feature selections improve OCC performance, we compare OCC methods on all sites. To perform multiple comparisons of multiple methods on multiple databases, we adopt the Friedman Aligned-Ranks test (García et al., 2010) and post-hoc Holm method (García et al., 2010) with a level of confidence of 0.05.

The Friedman test is used to detect significant differences among the performance measures of the methods studied. When there is a significant difference, we proceed with the post-hoc Holm procedure, to find out which algorithms are significantly different in terms of performance among the 1*n comparisons performed (García et al., 2010).

4.5 Results

Tables 1 and 2 shows the means of measures A and AUC of the different techniques tested. We also have Precision, F, and Recall results. However, due to space limitations, we do not list them here.

First of all, we need to highlight that svddd and som_dd are unable to find a data model with the available computational resources.

A descriptive analysis of the results leads to the following remarks: (i) The performance of OCC with feature selection is worse than OCC without this data preprocessing. For example if data-processing is done, measures A and AUC of gauss_dd are 0.75 and 0.80 respectively. (ii) It is clear that the gauss_dd method significantly outperforms parzen_dd and knndd algorithms. In 21 of the 27 sites its values of A and ACC are the best. (iii) The performance OCC is stable over all databases (the standard deviation is near 0). However, only in sites 6 and 26, are the values of A and ACC poor.

Table 3 shows that a normal distribution of AUC values cannot be assumed with 0.05 level of confidence. Because of this, we perform several non-parametric tests suggested in (García et al., 2010), throughout the results study.

The Wilcoxon Signed-Ranks Test (Table 4) shows that the OCC which used feature selection have a lower performance than methods that use all features. Therefore, we do not consider the use of feature selection methods in the rest of the experiment.

The first column of table 5 shows the ordered average ranks with AUC obtained by each method in the Friedman test. Gauss_dd is the OCC with best performance, so that is assigned to rank 1; knndd (the second best) to rank 2; and parzen_dd to rank 3.

The second column of table 5 shows that the adjusted p-values with the Holm post-hoc test are present. The Holm procedure rejects null hypotheses

Table 1: Table of results for OCC without feature selection.

	Parzen_dd		Knndd		Gauss_dd			Parzen_dd		Knndd		Gauss_dd	
Site	A	AUC	A	AUC	A	AUC	Site	A	AUC	A	AUC	A	AUC
1	0.95	0.93	0.71	0.78	0.96	0.97	16	0.71	0.75	0.88	0.91	0.97	0.98
2	0.95	0.93	0.80	0.85	1.00	1,00	17	0.21	0.55	0.60	0.77	0.77	0.87
3	0.62	0.78	0.71	0.83	0.88	0.93	18	0.62	0.75	0.93	0.95	0.94	0.96
4	0.63	0.71	0.93	0.95	0.97	0.98	19	0.88	0.88	0.75	0.75	0.99	0.99
5	0.98	0.98	0.75	0.75	1.00	1,00	20	0.99	0.99	0.94	0.96	0.95	0.96
6	0.50	0.50	0.52	0.52	0.52	0.52	21	0.25	0.50	0.89	0.93	0.89	0.93
7	0.37	0.62	0.92	0.96	0.93	0.96	22	0.85	0.85	0.75	0.75	0.99	0.99
8	0.26	0.51	0.83	0.89	0.84	0.90	23	0.61	0.72	0.79	0.86	1.00	1,00
9	0.50	0.50	1.00	1,00	1.00	1,00	24	0.97	0.97	0.77	0.77	1.00	1,00
10	0.33	0.50	0.95	0.96	0.96	0.97	25	0.75	0.78	0.68	0.76	0.88	0.91
11	0.99	0.98	1.00	1,00	1.00	1,00	26	0.84	0.84	0.69	0.69	0.75	0.75
12	0.79	0.84	0.72	0.81	0.88	0.92	27	0.95	0.92	0.56	0.67	1.00	1,00
13	0.75	0.75	0.64	0.76	0.84	0.89							
14	0.60	0.71	0.60	0.75	0.76	0.85	Means	0.69	0.76	0.78	0.83	0.91	0.93
15	0.72	0.72	0.75	0.75	1.00		Std Dev	0.06	0.03	0.02	0.01	0.01	0.01

Table 2: Results table for OCC with feature selection.

	Parzen_dd		Knndd		Gauss_dd			Parzen_dd		Knndd		Gauss_dd	
Site	A	AUC	A	AUC	A	AUC	Site	A	AUC	A	AUC	A	AUC
1	0.90	0.90	0.45	0.59	0.49	0.62	16	0.72	0.77	0.64	0.73	0.78	0.83
2	0.99	0.98	0.73	0.79	0.74	0.81	17	0.21	0.55	0.51	0.72	0.75	0.86
3	0.52	0.72	0.59	0.76	0.78	0.87	18	0.62	0.75	0.81	0.87	0.81	0.87
4	0.63	0.71	0.73	0.82	0.74	0.83	19	0.91	0.91	0.72	0.72	0.72	0.72
5	0.57	0.57	0.74	0.74	0.74	0.74	20	0.99	0.99	0.92	0.94	0.85	0.89
6	0.50	0.50	0.50	0.50	0.50	0.50	21	0.25	0.50	0.89	0.93	0.89	0.93
7	0.37	0.62	0.93	0.96	0.92	0.96	22	0.88	0.88	0.75	0.75	0.75	0.75
8	0.26	0.51	0.70	0.80	0.80	0.87	23	0.60	0.72	0.75	0.83	0.88	0.92
9	0.50	0.50	1.00	1,00	1.00	1,00	24	0.99	0.99	0.76	0.76	0.82	0.82
10	0.33	0.50	0.87	0.90	0.82	0.86	25	0.67	0.74	0.57	0.67	0.77	0.83
11	0.62	0.74	0.93	0.96	0.97	0.98	26	0.88	0.88	0.53	0.53	0.58	0.58
12	0.70	0.78	0.62	0.75	0.63	0.75	27	0.80	0.85	0.51	0.63	0.60	0.70
13	0.55	0.66	0.49	0.66	0.63	0.75							
14	0.35	0.59	0.52	0.70	0.52	0.70	Means	0.62	0.72	0.70	0.77	0.75	0.80
15	0.56	0.56	0.75	0.75	0.75	0.75	Std Dev	0.06	0.03	0.03	0.02	0.02	0.01

Table 3: *p*-Lilliefors (Kolmogorov-Smirnov) values Normality test.

	Without Feature Selection		
	Gauss_dd	Parzen_dd	Knndd
	AUC	AUC	AUC
p-values	1.587e-12	8.318e-19	1.041e-10

	Feature Selection		
	Gauss_dd	Parzen_dd	Kndd
	AUC	AUC	AUC
p-values	2.996e-19	5.750e-21	3.240e-09

(the significantly different in AUC do not exit) that have a *p*-value lower than 0.05.

An analysis of these results leads to remarks that gauss_dd is the method which has the best average AUC and knndd and parzen_dd has been outperformed with this level of significance.

Table 4: *p*-values of Wilcoxon's test. Comparing algorithms without feature selection against methods with feature selection.

	Equal	Worst	Best
	AUC	AUC	AUC
Gauss_dd	7.43e-10	3.71e-10	**1**
Parzen_dd	0.0090	0.0045	**0.995**
Knndd	2.73e-09	1.36e-09	**1**

Table 5: Average algorithm rankings Aligned Friedman) and adjusted Holm test *p*-values.

Algorithm	FA Ranking	Holm APV
Gauss_dd	160.3299	–
Knndd	260.9175	0.000031
Parzen_dd	345.6495	0

4.6 Discussion

We have evaluated 5 OCC techniques in an Information Extractor verifier database. The performances of

the methods have been statistically compared using Wilcoxon, Friedman and Holm tests. From our experiments results we make 3 observations: (i) Feature Selection techniques gave poor performances. The reason is that the Laplacianan Score selected all categorical features and very few categorical features. Hence, the dissimilarity measured applied by each OCC cannot be calculated correctly with certain types of data. (ii) In a few sites, OCC did not work well because our characteristics set does not contain any feature that discriminates against these classes. (iii) Gauss_dd was the best method for almost all sites because feature values fit closely to a normal distribution.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we discussed OCC techniques for solving Information Extractor verify problems. Five basic OCC methods were studied. A comprehensive evaluation of these methods was conducted to compare their performances which enable us to conclude that Gauss_dd outperforms all the testing techniques.

Still, there are several problems that are open for research. Feature database and pre-processing phases have not been exploited very much for our problem. Another point to note here is that classifier ensembles and other sophisticated OCC like SVM or Bayesian Network approach have not been investigated. Also, data complexity measures would be an interesting exercise, if we want a quick way to choose an OCC yielding good performance for a particular site.

ACKNOWLEDGEMENTS

This work is supported by the European Commission (FEDER), the Spanish and the Andalusian R&D&I programmes (grants TIN2007-64119, P07-TIC-2602, P08-TIC-4100, TIN2008-04718-E, TIN2010-21744, TIN2010-09809-E, TIN2010-10811-E, and TIN2010-09988-E).

REFERENCES

Bernstein, P. A. and Haas, L. M. (2008). Information integration in the enterprise. *Commun. ACM*, 51(9):72–79.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3).

Chidlovskii, B., Roustant, B., and Brette, M. (2006). Documentum eci self-repairing wrappers: performance analysis. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 708–717, New York, NY, USA. ACM.

Demsar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

García, S., Fernández, A., Luengo, J., and Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, 180(10):2044–2064. Special Issue on Intelligent Distributed Information Systems.

Hempstalk, K., Frank, E., and Witten, I. H. (2008). One-class classification by combining density and class probability estimation. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, ECML PKDD '08, pages 505–519, Berlin, Heidelberg. Springer-Verlag.

Hodge, V. J. and Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22:2004.

Kushmerick, N. (2000). Wrapper induction: Efficiency and expressiveness. *Artificial Intellligence*, 118(1-2):15–68.

Lerman, K., Minton, S. N., and Knoblock, C. A. (2003). Wrapper maintenance: A machine learning approach. *Journal of Artificial Intelligence Research*, 18:2003.

Madhavan, J., Cohen, S., Halevy, A. Y., Jeffery, S. R., Dong, X. L., Ko, D., and Yu, C. (2007). Web-scale data integration: You can afford to pay as you go. In *CIDR*, pages 342–350.

McCann, R., AlShebli, B., Le, Q., Nguyen, H., Vu, L., and Doan, A. (2005). Mapping maintenance for data integration systems. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1018–1029. VLDB Endowment.

Tax, D. (2009). Ddtools, the data description toolbox for matlab. version 1.7.3.

Tax, D. M. J. (2001). *One-class classification, concept learning in the absence of counter example*. PhD thesis, Delft University of Technology.

Villalba, S. D. and Cunningham, P. (2007). An evaluation of dimension reduction techniques for one-class classification. *Artificial Intelligence Review*, 27(4):273–294.

Weiss, J. (2005). Aligning relationships: Optimizing the value of strategic outsourcing. Technical report, IBM.