Neural Comput & Applic (2017) 28 (Suppl 1):S699–S705
DOI 10.1007/s00521-016-2374-9

ORIGINAL ARTICLE

Malicious URL detection via spherical classification

A. Astorino[1] · A. Chiarello[1] · M. Gaudioso[2] · A. Piccolo[2]

Received: 1 April 2015 / Accepted: 21 May 2016 / Published online: 7 June 2016

Abstract We introduce and test a binary classification method aimed at detecting malicious URL on the basis of some information on both the URL syntax and its domain properties. Our method belongs to the class of supervised machine learning models, where, in particular, classification is performed by using information coming from a set of URL's (samples in machine learning parlance) whose class membership is known in advance. The main novelty of our approach is in the use of a spherical separation-based algorithm, instead of SVM-type methods, which are based on hyperplanes as separation surfaces in the sample space. In particular we adopt a simplified spherical separation model which runs in $O(tlogt)$ time (t is the number of samples in the training set), and thus is suitable for large-scale applications. We test our approach using different sets of features and report the results in terms of training correctness according to the well-established tenfold cross-validation paradigm.

Keywords Classification · Spherical separation · Malicious Web sites

✉ M. Gaudioso
gaudioso@dimes.unical.it

A. Astorino
astorino@icar.cnr.it

A. Chiarello
chiarello@icar.cnr.it

A. Piccolo
piccolo@dimes.unical.it

[1] ICAR–CNR, c/o University of Calabria, 87036 Rende, CS, Italy

[2] DIMES, University of Calabria, 87036 Rende, CS, Italy

1 Introduction

A useful resource to prevent risks in computer security is provided by the so-called black lists, which are databases containing a typically large number of IP addresses, domain names and related URL's for suspicious sites in terms of generation of threats. A rich literature is available on the creation and usage of such lists (see, e.g., [17]). If a URL is comprised into a black list, it is convenient to deviate the network traffic from it and, in fact, many internet service providers (ISP) simply block all messages coming from it. The users who detect anomalies in messages or activities they consider suspicious often transfer the related information to appropriate Web sites devoted to risk analysis.

Other possible way to compile black lists is the use of certain spam trap addresses which are diffused in the aim of being contacted by crawler spiders, typically used by phishers. As soon as one of such site address is contacted, the calling site is included into the black list.

Although black lists are rather useful, we cannot expect that they are exhaustive of all possible threats, either because the number of potentially dangerous site is extremely high or because the system is highly dynamic and it is almost impossible to keep any black list sufficiently updated.

Every time there exists any suspect about the malicious nature of a site the Whois service is able to provide some useful information in terms of IP, domain name and other characteristics related to it. Whois registers are publicly available, and there exist online services providing upon request such information, in an appropriate form.

The very basic idea of [10] is to use the information available about a given set of URL's, in connection to the related Whois, to design a classifier based on some

machine learning technique. In particular one of the tools adopted is the well-known SVM paradigm which, being suitable for supervised classification, requires that a sufficiently rich training set is available in advance. Such training set is constituted by a list of URL's labeled in the form malicious–nonmalicious. A set of both qualitative and quantitative features is defined, and each URL is associated with a string of possible values of the features.

Of course different sets of features can be adopted for classification purposes, and in next section, we will describe in details the ones we have considered.

The differences of our approach w.r.t. the one previously cited [10] are twofold. As we are aimed at providing a methodology suitable for very large datasets too, we have confined ourselves to a limited number of features (e.g., we have used no "bag of words," which in general requires an explosion in the size of the sample space) and, on the other hand, we have adopted a low complexity algorithm, accepting in advance the possibility of obtaining less accurate classification performance w.r.t. the SVM approach which requires solution of a structured quadratic programming problem [12].

Following [10] we take into account in our classification model both lexical features of the URL and host information, as those provided by the Whois. As a classification tool, we adopt the spherical separation paradigm [3, 4], which differs from SVM basically because separation in the feature space is not pursued by means of a hyperplane, instead by a spherical surface (application of ellipsoidal surfaces has been introduced too in [1]). Of course goodness of the classification tool depends on the geometry of the sets to be separated and cannot be easily predicted. Our choice for spherical separation has been dictated by the availability of a simplified approach to it [2], where the center of the separating sphere is calculated in advance and only the radius is optimized. Such approach allows us to calculate the classifier in $O(t \log t)$, where t is the size of the training set and, consequently, appears suitable for dealing with large-scale applications.

The paper is organized as follows. In Sect. 2 we introduce our classification model, describing in details the different set of features we have adopted. In Sect. 3 we summarize the spherical separation algorithm we have implemented. In Sect. 4 we report the results of our computational experience on some benchmark training sets drawn from the literature, together with some conclusions.

2 The classification model

We present now the list of seven features that we used in our work to detect malicious URLs. For our purposes, we decided to not analyze the URL's page structure or its content: all the features have been generated using information derived from the general URL syntax (every URL consists of the following three parts: $<\texttt{protocol}> : // <\texttt{hostname}> / <\texttt{path}>$). Some of the features used are strictly related to the lexical properties of the URL (intended as a character string) and other to the properties of the URL's hostname (available thanks to a Whois query).

The features taken into consideration are:

1. *Number of subdomains* This is the count of the subdomains which are detectable in the text of the URL. We do not consider only the URL's hostname but also the URL's path which is often used to redirect users to other dangerous Web sites.
2. *URL age* We consider the age in days of the URL's hostname (how long its content has been on the web). This is one of the most important features in malicious URL detection problem because dangerous sites usually have a very short life and are recently registered compared to safe sites. The registration date of URL's hostname is needed to calculate the value of this feature.
3. *URL expiration* We count the number of days remaining before the expiration date of URL's hostname. Also this feature is potentially useful since dangerous sites usually are registered for shorter time than safe ones.
4. *Hostname length* This is the simply count of the textual characters forming the URL's hostname.
5. *Path length* This is the simply count of the textual characters forming the URL's path. Excessive length is often correlated with suspicious re-addressing.
6. *IP address geographical location* IP addresses related to dangerous sites are usually located in specific geographical areas so we use this feature to express the information about the country location of the IP address related to the URL's hostname. The list of countries used to evaluate this parameter has been reduced to just ten nations: Canada, China, Finland, France, India, Italy, Spain, Turkey, UK and USA. All other countries have been bundled together into a unique term, for the IP address located in countries not included in our list. This is a categorical feature.
7. *Presence or not of the word "Login"* Dangerous sites usually contain in the text of their web address specific terms in order to cheat the web users. Thus, we have included such binary feature to report possible presence of the word "Login" in the text of the URL.

According to their definition, all the features taken into consideration (but "IP address geographical location") can assume integer values greater than or equal to zero. As the number of features we have adopted is small, we have performed several experiments using different sets of

features. According to the number of features used, the URL is represented in a different vector format. Using different sets of features it is important to detect the relevant ones and to discard those unable to provide any significant contribution to the classification process.

The malicious URLs detection problem has been modeled as a binary classification problem, and the predefined classes of URLs have been two: the set of malicious URLs (including URLs dangerous for the web users) and the set of benign URLs (including URLs safe for the web users).

3 Spherical separation

In many supervised machine learning problems the objective is to assign elements to a finite set of classes or categories. For a given set of sample points coming from two classes, we want to construct a function for discriminating between the classes. The goal is to select a function that will efficiently and correctly classify future points. Classification techniques can be used for data mining or pattern recognition, where many applications require a categorization.

The classical binary classification problem is to discriminate between two finite sets of points in the n-dimensional space, by a separating surface. The problem consists in finding a separating surface minimizing an appropriate measure of the classification error.

Several mathematical programming-based approaches for binary classification have been historically proposed [5, 11, 15]. Among the more recent ones we recall the support vector machine (SVM) technique [6, 16], where a classifier is constructed by generating a hyperplane far away from the points of the two sets. By adopting kernel transformations within the SVM approach, we can obtain general nonlinear separation surfaces. In this case the basic idea is to map the data into a higher-dimensional space (the feature space) and to separate the two transformed sets by means of one hyperplane that corresponds to a nonlinear surface in the original input space.

Parallel to the development of SVM methods, the use of nonlinear separating surfaces in the dataset space, instead of hyperplanes, has received in recent years some attention. In particular, in our work, we have considered the spherical separation approach, characterized by the fact that the separation process takes place in the original input space and does not require mapping to higher dimension spaces.

More formally, let

$$\mathcal{X} = \{x_1, \ldots, x_p\}$$

be a set of samples (or points) $x_i \in \mathbb{R}^n$. In the supervised learning, we assume that, in correspondence to any point x_i of $\mathcal{X}$, a label y_i is given. The case $y_i \in \mathbb{R}$ is known as "regression," while, when the label y_i takes values in a discrete finite set, the task is a "classification" process. A particular case of the latter is the binary classification, where, for each i, the label y_i can assume only two possible values. The objective of the supervised learning is to predict the label of any new sample only on the basis of the information of the labeled points (the training set).

In particular, in the binary classification problems, we consider the following partition of $\mathcal{X}$ into two nonempty sets:

$$\mathcal{X}_+ := \{(x_i, y_i) | x_i \in \mathbb{R}^n, y_i = +1, \quad i = 1, \ldots, m\}$$

and

$$\mathcal{X}_- := \{(x_i, y_i) | x_i \in \mathbb{R}^n, y_i = -1, \quad i = m+1, \ldots, p\}.$$

In the spherical separation approach we define the set $\mathcal{X}_+$ *spherically separable* from $\mathcal{X}_-$ if and only if there exists a sphere

$$S(x_0, R) := \{x \in \mathbb{R}^n | (x - x_0)^T (x - x_0) \leq R^2\}$$

centered in $x_0 \in \mathbb{R}^n$ of radius $R \in \mathbb{R}$, such that

$$y_i(\|x_i - x_0\|^2 - R^2) \leq 0 \quad i = 1, \ldots, p. \tag{3.1}$$

In addition, when inequalities (3.1) are strictly satisfied, then the sets $\mathcal{X}_+$ and $\mathcal{X}_-$ are strictly spherically separated, i.e.,

$$\begin{aligned} &\|x_i - x_0\|^2 \leq (R - M)^2 \quad i = 1, \ldots, m \\ &\|x_i - x_0\|^2 \geq (R + M)^2 \quad i = m+1, \ldots, p, \end{aligned} \tag{3.2}$$

for some M, the margin, such that $0 < M \leq R$. Setting $q := 2RM$ and $r := R^2 + M^2$, inequalities (3.2) become:

$$q + y_i(\|x_i - x_0\|^2 - r) \leq 0 \quad i = 1, \ldots, p.$$

In general it is not easy to know in advance whether the two sets are strictly spherically separable; then, in [2–4, 9] a classification error function has been defined in order to find a minimal error separating sphere.

In particular, in [2, 3] a separating sphere has been obtained by minimizing the following objective function:

$$z_{S_1}(x_0, r) = r + C \sum_{i=1}^{p} \max\{0, y_i(\|x_i - x_0\|^2 - r)\},$$

with $M = 0$ and $r \geq 0$. C is a positive parameter giving the trade-off between the minimization of the radius and the minimization of the classification error.

More precisely in [2] an ad hoc algorithm that finds the optimal solution in $O(t \log t)$, with $t = \max\{m, p - m\}$, has been presented for the case where the center x_0 is fixed. It basically consists of two phases: the sorting phase and the cutting phase. In the first one the sample points are sorted according to their distance from the center, while in the

second one an optimal cut is found. The adopted simplification is rather drastic; nevertheless, a judicious choice of the center (e.g., the barycenter of the set $\mathcal{X}_+$) has allowed to obtain reasonably good separation results at a very low computational cost.

In [3] the spherical separation problem has been tackled without any constraint in the location of the center by means of DC algorithm (DCA) [8, 13], based on a difference of convex (DC) decomposition of the objective function. A similar DCA approach has been proposed in [9] for minimizing the following error function:

$$z_{S_2}(x_0, r) = r^2 + C \sum_{i=1}^{p} \max\{0, y_i(\|x_i - x_0\|^2 - r)\}^2,$$

with $M = 0$. The choice of z_{S_2} instead of z_{S_1} is motivated by the fact that using z_{S_2} allows a DC decomposition where all the computations in DCA are explicit.

Finally the DCA scheme has been used also in [4], for minimizing the following error function:

$$z_{S_3}(x_0, q, r) = C \sum_{i=1}^{p} \max\{0, y_i(\|x_i - x_0\|^2 - r) + q\} - q,$$

with $0 \le q \le r$ and $M \ge 0$, where the term $-q$ is aimed at maximizing the margin. Moreover, similarly to [2], also in [4] an ad hoc algorithm that finds the optimal solution in $O(t \log t)$ has been designed when, in function z_{S_3}, the center x_0 is fixed.

4 Computational experience

We present now the results of our numerical experiments in malicious URLs detection. We have adopted the spherical classification approach described in Sect. 3. In particular we have coded in MATLAB (on a Pentium V 2.60 GHz Notebook) the algorithm introduced in [2], which is able to find the optimal solution in case the center of the separating sphere is fixed in advance.

For evaluating the effectiveness of our approach we have used a dataset obtained as follows. The samples have been randomly selected using some sources available on the web; in particular, for malicious URLs we used the online black list called PhishTank [14], a free community site where anyone can submit, verify, track and share phishing data. As for benign URLs we have collected our samples by developing a fast web crawler (a program that visits web pages and collects information) capable of gathering unique URLs (not duplicated) specifying the minimum and maximum length of the URL. The crawling phase started from the DMOZ Open Directory Project [7], a human-edited directory of safe web pages. The size of our dataset is 245 malicious and 384 benign URLs (dataset 1).

We have applied the standard tenfold cross-validation protocol. The dataset has been divided into ten subsets of equal size, and each subset has been validated using the classifier obtained by training executed on the remaining nine subsets. The classification accuracy is defined as the average percentage of well-classified URLs (of both classes).

We have run the program for different values of the positive weighting parameter C. We have implemented two possible choices of the center x_0, by selecting, respectively, the barycenter of the set of malicious URLs ($x_0^{(1)}$) and the barycenter of the set of benign URLs ($x_0^{(2)}$). However, after our evaluation, we have noticed that setting the center x_0 equal to the barycenter of the set of malicious URLs was the most performing choice.

We have executed first an exhaustive feature selection process on five out of the seven features analyzed in Sect. 2: number of subdomains, URL age, URL expiration, hostname length and path length (Table 1). In Table 2 we report the most significant results obtained and the corresponding set of features. In Table 3 the true positive rate and the true negative rate for the best performing combination of features are reported, where the true positive rate is defined as the percentage of benign URLs correctly classified as benign URLs and the true negative rate as the percentage of malicious URLs correctly classified as malicious URLs.

Table 1 Features description

Feature number	Description
1	Number of subdomains
2	URL age
3	URL expiration
4	Hostname length
5	Path length

Table 2 Average correctness on dataset 1

Feature combination	Parameters	Average training set correctness (%)	Average testing set correctness (%)
2, 3	$C = 0.1$, $x_0^{(1)}$	86.3	86.3
2	$C = 10$, $x_0^{(1)}$	85.6	85.0
1, 2, 3	$C = 0.1$, $x_0^{(1)}$	83.7	83.5
2, 3, 4	$C = 0.1$, $x_0^{(1)}$	82.8	82.2

The results of the feature selection process indicate that the best classification accuracy is obtained by considering a quite small subset of the available features, in particular the combination of the URL age and URL expiration features.

Table 3 Average true positive and true negative rate on dataset 1

	Average training set true positive (%)	Average training set true negative (%)	Average testing set true positive (%)	Average testing set true negative (%)
Features: 2, 3 $C = 0.1$, $x_0^{(1)}$	90.3	80.1	90.3	80.0
Features: 2 $C = 10$, $x_0^{(1)}$	88.2	81.5	88.0	80.4
Features: 1, 2, 3 $C = 0.1$, $x_0^{(1)}$	88.1	76.6	87.7	76.7
Features: 2, 3, 4 $C = 0.1$, $x_0^{(1)}$	87.4	75.5	87.0	74.7

Table 4 Average correctness on dataset 2

Feature combination	Parameters	Average training set correctness (%)	Average testing set correctness (%)
2, 3	$C = 0.1$, $x_0^{(1)}$	83.9	83.9
2, 5	$C = 10$, $x_0^{(1)}$	83.4	83.3
2	$C = 10$, $x_0^{(1)}$	83.6	83.4
2, 3, 5	$C = 10$, $x_0^{(1)}$	83.3	83.2

The remaining ones are redundant and, sometimes, even misleading.

We have extended the numerical experiments applying our methodology to a bigger dataset constituted by 11,975 URLs, 5090 malicious and 6885 benign (dataset 2). It has been constructed by adopting the same methodology as for dataset 1. The results are shown in following Tables 4 and 5. Also in this case we report the subsets of features which have provided better results.

We have performed some additional experiments on both datasets by considering two more features: the IP address geographical location and the presence of the "suspicious" word *Login* in the text of the URL (Table 6). Since such information was not available for all samples, the size of the datasets has been reduced to 370 benign URLs and 210 malicious ones for dataset 1 and to 6432 benign URLs and 4520 malicious ones for dataset 2.

Table 6 Features description

Feature number	Description
6	IP address geographical location
7	Presence or not of the word "Login"

Table 7 Additional experiments: average correctness on dataset 1

Feature combination	Parameters	Average training set correctness (%)	Average testing set correctness (%)
2, 7	$C = 0.1$, $x_0^{(1)}$	79.3	78.6
2, 5, 7	$C = 10$, $x_0^{(1)}$	76.8	77.1
2, 5, 6	$C = 10$, $x_0^{(1)}$	75.7	75.7
2, 3, 6	$C = 0.1$, $x_0^{(1)}$	75.6	74.5

As for dataset 1, in Table 7 we report the most significant results obtained for this series of experiments and in Table 8 the true positive and true negative rate for the best performing combination of features.

Table 5 Average true positive and true negative rate on dataset 2

	Average training set true positive (%)	Average training set true negative (%)	Average testing set true positive (%)	Average testing set true negative (%)
Features: 2, 3 $C = 0.1$, $x_0^{(1)}$	86.1	80.9	86.0	81.0
Features: 2, 5 $C = 10$, $x_0^{(1)}$	85.6	80.5	85.6	80.2
Features: 2 $C = 10$, $x_0^{(1)}$	85.7	80.7	85.8	80.3
Features: 2, 3, 5 $C = 10$, $x_0^{(1)}$	85.5	80.3	85.5	80.2

Table 8 Additional experiments: average true positive and true negative rate on dataset 1

	Average training set true positive (%)	Average training set true negative (%)	Average testing set true positive (%)	Average testing set true negative (%)
Features: 2, 7 $C = 0.1$, $x_0^{(1)}$	85.5	68.6	84.3	68.6
Features: 2, 5, 7 $C = 10$, $x_0^{(1)}$	81.9	67.7	82.4	67.6
Features: 2, 5, 6 $C = 10$, $x_0^{(1)}$	81.1	66.3	83.0	62.9
Features: 2, 3, 6 $C = 0.1$, $x_0^{(1)}$	82.5	63.6	84.0	57.6

Table 9 Additional experiments: average correctness on dataset 2

Feature combination	Parameters	Average training set correctness (%)	Average testing set correctness (%)
2, 6	$C = 0.1$, $x_0^{(1)}$	84.4	84.4
2, 7	$C = 0.1$, $x_0^{(1)}$	84.0	84.0
2, 3, 6	$C = 0.1$, $x_0^{(1)}$	84.1	84.0
2, 5, 6	$C = 10$, $x_0^{(1)}$	83.6	83.6

The above results show that the two additional features did not provide any positive contribution in terms of classification accuracy. We remark the role played by the URL age in this setting too.

As far as dataset 2 is concerned, the results are reported in Tables 9 and 10. The introduction of the additional features has provided, in this case, a slight improvement.

We observe, finally, that our methodology has given a rather satisfactory quality in the classification process. It appears worth noting that, even for the larger dataset 2, the computation time has always been of the order of few

Table 10 Additional experiments: average true positive and true negative rate on dataset 2

	Average training set true positive (%)	Average training set true negative (%)	Average testing set true positive (%)	Average testing set true negative (%)
Features: 2, 6 $C = 0.1$, $x_0^{(1)}$	86.8	81.0	86.8	81.0
Features: 2, 7 $C = 0.1$, $x_0^{(1)}$	86.5	80.5	86.5	80.6
Features: 2, 3, 6 $C = 0.1$, $x_0^{(1)}$	86.6	80.6	86.5	80.5
Features: 2, 5, 6 $C = 10$, $x_0^{(1)}$	86.1	80.2	86.1	80.2

seconds, which suggests possible utilization of our method at least as a preliminary classification tool in view of future very large-scale applications.

Acknowledgments This work has been partially supported by Italian M.I.U.R. Programma Operativo Nazionale (PON) 2007–2013, Project "Protezione dei servizi digitali e di pagamento elettronico," PON03PE_00032_2.

References

1. Astorino A, Gaudioso M (2005) Ellipsoidal separation for classification problems. Optim Methods Softw 20(2–3): 261–270
2. Astorino A, Gaudioso M (2009) A fixed-center spherical separation algorithm with kernel transformations for classification problems. CMS 6(3):357–372
3. Astorino A, Fuduli A, Gaudioso M (2010) DC models for spherical separation. J Glob Optim 48(4):657–669
4. Astorino A, Fuduli A, Gaudioso M (2012) Margin maximization in spherical separation. Comput Optim Appl 53(2):301–322
5. Bennett KP, Mangasarian OL (1992) Robust linear programming discrimination of two linearly inseparable sets. Optim Methods Softw 1:23–34
6. Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge
7. http://opendirectory.org/
8. Le Thi HA, Pham Dihn T (2005) The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. Ann Oper Res 133:23–46
9. Le Thi HA, Le HM, Pham Dinh T, Van Huynh N (2013) Binary classification via spherical separator by DC programming and DCA. J Glob Optim 56:1393–1407
10. Ma J, Saul LK, Savage S, Voelker GM (2009) Beyond blacklists: learning to detect malicious web sites from suspicious URLs. KDD'09, June 28–July 1, 2009. France, Paris, pp 1245–1253
11. Mangasarian OL (1965) Linear and nonlinear separation of patterns by linear programming. Oper Res 13:444–452
12. Palagi L, Sciandrone M (2005) On the convergence of a modified version of SVM^{light} algorithm. Optim Methods Softw 20(2–3): 317–334
13. Pham Dinh T, Le Thi HA (1998) A D.C. optimization algorithm for solving the trust-region subproblem. SIAM J Con Opt 8:476–505
14. PhishTank: http://www.phishtank.com/
15. Rosen JB (1965) Pattern separation by convex programming. J Math Anal Appl 10:123–134
16. Vapnik V (1995) The nature of the statistical learning theory. Springer, New York
17. Zhang J, Porras P, Ullrich J (2008) Highly predictive blacklisting. USENIX Security Symposium 2008—usenix.org