Lecture Notes in Management Science (2015) Vol. 7, 50–55 ISSN 2008-0050 (Print), ISSN 1927-0097 (Online)

Fraudulent URL Classification with an RBFNN

Dirk Snyman [1], Tiny du Toit [1] and Hennie Kruger [1]
[1] North-West University, Potchefstroom Campus, Potchefstroom, South Africa
{dirk.snyman, tiny.dutoit, hennie.kruger}@nwu.ac.za

Proc. ICAOR 2015
Vienna, Austria

Keywords:
Neural networks
Artificial intelligence
Machine learning

Abstract

Phishing attacks are a social engineering scam which aim to steal sensitive information like personal, financial and social details, from unsuspecting consumers. It is based on the premise that a significant number of consumers are ignorant of the technical security practices in a digital environment. These attacks establish trust from the user in order to lull them into a false sense of familiarity in which they are more likely to freely supply their identifying information. In this study an intelligent approach to phishing detection is presented. The method is based on the information contained in the Uniform Resource Locator (URL) that points to a phishing website. These URLs are analysed and classified as malicious or safe, by a new automated Radial Basis Function Neural Network (RBFNN) construction algorithm. This technique determines the best neural network architecture for a specific classification task and uses an in-sample model selection criterion. Example URLs which were collected from real world repositories, Open Directory and Phishtank, are used to train and evaluate the neural network. The results of a cross-validation experiment setup are presented. It was found that the RBFNN algorithm outperforms a naïve Bayes classifier baseline which is considered to be a standard text classification baseline in literature due to its simple structure and linear execution.

Introduction

In a range of publications of information security analysis over an extended period, Richardson (2010; 2008) and Berger (2012) state that viruses and malware account for the most commonly seen type of malicious intrusions noted in the corporate environment. Of the surveyed users, 49% reported incidents leading to financial losses and 67.1% reported information losses. These reported incidents extend beyond the corporate environment and also include personal losses. This poses a great risk to companies and individuals in the digital domain. Human malicious actions (in terms of wilful information security breaches or exploits from within the company) accounts for a much smaller fraction of the reported incidents of security violations. Richardson (2010; 2008) and Berger (2012) report that only 3 percent of the losses are attributed to malicious intent by insiders. The losses due to human negligence or error, without the intent being to breach information security, is reported to be at 14.5%; a larger number than malicious intent and arguably one of the most preventable issues.

One type of malicious endeavour that relies on and specifically exploits human error and ignorance is phishing (APWG, 2014; Jacobson, 2007; Dhamija *et al.*, 2006). The Anti-Phishing Working Group (2014) describes phishing as a malicious endeavor that aims to steal sensitive information like personal, financial and social details, from unsuspecting consumers by employing social engineering approaches in a technical environment. These approaches are usually based on fraudulent electronic messages that are sent to consumers which are presented in such a manner that they seem to be originating from a trusted supplier, institution, or other service that the consumer has an affiliation with. These messages usually contain an appeal to the consumer to follow a link to a website, mimicking that of the specific trusted institution. They are then coerced into supplying personal and financial information to the fraudulent website which is subsequently unlawfully used for a range of criminal activities, including identity theft and financial exploitation.

According to Dhamija *et al.* (2006), traditional indicators that could expose the abovementioned websites as being fraudulent are failing due to users being either inexperienced or ignorant about these indicators, or due to knowledgeable users being so convinced with the visual similarity to the site they know and trust, that they neglect to confirm the site's validity. The mitigation of human error can be achieved through the implementation and strict application of policies and guidelines that govern human interaction with sensitive information. Policies however are only effective as long as the implementation thereof is actively monitored and enforced and even then these policies do no guarantee safety but merely limit the factor of possible human error that could lead to information losses (Cranor, 2008).

In an attempt to limit the human factor in the detection of a phishing website, this paper presents a machine learning approach to identify whether a site would be safe to visit, or should be avoided, based on information contained in the

Uniform Resource Locator (URL) that points to the relevant resource. The classification of a site being either malicious or benign will be achieved by employing a Radial Basis Function Neural Network (RBFNN). The RBFNN structure will be determined by means of a construction algorithm called BH-RBFNN (Best Hidden RBFNN). The resulting RBFNN algorithm's performance on a real world dataset will be evaluated by comparing it to a naïve Bayes algorithm as the baseline.

The remaining sections of the paper are structured as follows: the implementation of an RBFNN and the architecture determination thereof are discussed; the experimental setup for this research, including the URL deconstruction and representation, is explained and finally the results and conclusion to this research are presented.

The Radial Basis Function Neural Network

Artificial neural networks (ANNs) are a commonly used method for intelligent systems as ANNs can classify unknown situations based on, and inferred from a collection of similar observations (Dahl *et al*., 2013; Shen *et al*., 2011; El Bakry, 2010). Broomhead and Lowe (1988) postulated an adaptation of the standard ANN to use a Radial Basis Function as the activation function to perform interpolation in a high-dimensional space. The Radial Basis Function Neural Network (RBFNN) became a popular method for the use in intelligent systems that require function approximation or learning problems that yield predictions with non-linear models. An RBFNN learns patterns in the data in a manner similar to finding (describing) a function that is able to model a multi-dimensional dataset. The model is described in such a way that it provides a statistically best fit (Manjunatha *et al*., 2012). In other words, it is able to model any continuous function which makes it flexible enough to be implemented for almost any problem. When a model is fully trained it delivers fast and efficient performance and is easily implemented for example in browser plugins.

The RBFNN function is mathematically represented as follows:

$$E(y) = \sum_{j=1}^{M} w_j \Phi_j(X) + w_0 \,. \tag{1}$$

The expected value of the target *E(y)* is a summation of basis functions, where w_j is the assigned weight to the *jth* radial basis function, $\Phi_j(X)$ denotes the radial basis function itself, and w_0 denotes the bias incorporated in the model. The most popular radial basis function used for RBFNNs is the Gaussian (2) where X is a d-dimensional input vector, μ_j is the center of the basis function and v_j is the width of the basis function:

$$\Phi_j(X) = \exp\left(-\frac{\|X - \mu_j\|^2}{2v_j^2}\right). \tag{2}$$

The underlying architecture of an RBFNN plays a vital role in the performance and suitability for a specific application (Manjunatha *et al*., 2012; Shen *et al.*, 2011). An RBFNN architecture consists of three layers, being an input-, hidden-, and output layer. These are comprised of N input neurons, H hidden neurons and M output neurons respectively. The determination of a specific architecture is, however, no arbitrary task as there are different levels of the architecture to be optimised, namely the number of hidden nodes and the weights to be assigned to and from each hidden node. An exhaustive search through this instance space of options is time consuming and ineffective. The automatic determination thereof is needed to ensure optimality of the system while reducing time and promoting effectiveness (Manjunatha *et al*., 2012; Broomhead & Lowe, 1988).

To facilitate the automatic determination of the RBFNN architecture the Best Hidden RBFNN (BH-RBFNN) algorithm (Du Toit & Hamadneh, 2013) is employed. The BH-RBFNN algorithm automatically determines the best architecture by simulating a trial-and-error approach. The algorithm was derived from a cross-validation based neural network construction algorithm described by Setonio (2001). This algorithm is presented in Algorithm 1.

The RBFNN algorithm commences with a simple RBFNN model ($\mathcal{N}_1$) that is trained with a single hidden neuron (H). This model is trained to minimize the Root Mean Squared Error (RMSE) and a model selection criterion value SBC – Schwarz Bayesian Criterion (Scwarz,1978) as implemented by Du Toit & Hamadneh (2013)—is assigned to the model. The process is repeated for a second RBFNN model, $\mathcal{N}_2$, but the number of hidden neurons (H) of $\mathcal{N}_2$ is set to the number of hidden neurons for $\mathcal{N}_1$, incremented by a predetermined value h. The SBC-values of these models are compared and if $SBC_{\mathcal{N}_2}$ is less than $SBC_{\mathcal{N}_1}$ *(i.e. the* $\mathcal{N}_2$ denotes a better network than $\mathcal{N}_1$), $\mathcal{N}_1$ is replaced by $\mathcal{N}_2$ and the process continues with hidden neurons $H+h$, effectively growing the network. For each iteration the best parameters found in the preceding iteration are used as the initial parameters for the subsequent iteration.

The model is then grown until a stopping criteria is met ($H = maxH$) or the change in SBC value does not show any improvement. In the latter case the same RBFNN architecture is trained from random starting parameters and only if the SBC value does not again improve the algorithm terminates. The following section describes the experimental setup used for this research.

Algorithm 1: BH-RBFNN algorithm

```
Input: Input Features(N), Output Features(M), Hidden Neurons (H=1), Hidden Neuron Incrementation(h)

Output: Optimised RBFNN architecture

function BH-RBFNN(N,M,H,h) returns RBFNN

 I.  N1 = new RBFNN(N, H, M);
 II. init parameters of N1 randomly;
       train N1(min. RMSE);
       SBCN1 = SBCvalue();
 III.N2 = new RBFNN(N, H+h, M);
 IV. train N2(min. RMSE);
       SBCN2 = SBCvalue();
 V.  if (SBCN2 < SBCN1) then
     |  H := H + h;
     |  N1 := N2
     |  SBCN1 := SBCN2;
     |  if (H < MaxH) then
     |  |  goto III;
     |  else
     |  N3 = new RBFNN(N, H+h, M);
     |  init parameters of N3 randomly;
     |     train N3(min. RMSE);
     |     SBCN3 = SBCvalue();
     |  if (SBCN3 < SBCN1) then
     |  |  H := H + h;
     |  |  N1 := N3
     |  |  SBCN1 := SBCN3;
     |  |  if H < MaxH then
     |  |  |  goto III;
 VI. return N1
```

Experimental setup

The training corpus used for this research consisted of 24000 instances representing a collection of 12000 benign and 12000 malicious URL examples. These URLs were obtained by crawling repositories of known malicious URL examples (Phistank) and offset against known benign URL examples (Open Directory). From this corpus, features were extracted as a normalised bag of words (Finn & Kushmerick, 2006) based on URL snippets. The features were constructed as follows: A hyperlink in an email can be set to display any text while still pointing towards a URL that is hidden from the user. For example the hyperlink displayed in a spam email can be similar to an innocent looking URL like https://www.absa.co.za while actually pointing to http://www.gapsa.co.za/login%4/user1024/#redir.aspx. The full intended address or destination of the hyperlink is then deconstructed to obtain an array of informative "words". These words are delimited by a collection of characters that logically compartmentalise the URL into snippets. This collection includes characters such as [. | , | / | \ | etc.]. The snippets were then added to a global dictionary where only unique entries are held, for example:

Dictionary = { www, gapsa, co, za, login%4, #user1024, redir, aspx, … }.

A binary vector representation (3), referencing the global dictionary was constructed for each URL in the training set resulting in a global dictionary size of approximately 50000 words:

$$\vec{x} = \langle x_1, x_2, \cdots, x_n \rangle. \tag{3}$$

The vector ($\vec{x}$) was constructed with elements $x_1, x_2 \ldots x_n$ representing a reference to the dictionary where the presence ($x = 1$) or absence ($x = 0$) of a snippet in each training instance is indicated.

Feature selection was then performed by ranking candidate attributes by mutual information (4) to determine which attributes contribute the most information to the classification task,

$$MI(X;C) = \sum_{C \in \{M,B\}, X \in \{0,1\}} P(X = x, C = c) \cdot \log \frac{P(X = x, C = c)}{P(X = x) \cdot P(C = c)}, \quad (4)$$

where X denotes the binary presence or absence $\{1, 0\}$ of an attribute and C is the target classes of the classification problem (M = Malicious, B = Benign).

Evaluation and results

The training data was used in a cross-validation experiment consisting of 10 folds. For each fold the training data was randomly divided into a 90:10 ratio of training vs. test data. With each iteration a new 90:10 split was chosen. The average results for all folds gave an indication of the generalisation ability of the algorithm and is indicative of the expected performance on unseen instances. This cross-validation was performed on different numbers of features included in the training data. The features were incremented with one extra feature per iteration. For the first iteration, the feature included was the feature that provided the best discriminatory information for both the classes and each added feature follows a descending order through features ranked by mutual information. Classification performance was measured in terms of the accuracy of the classifier. In other words, how many of the unknown instances classified by the algorithm have been classified correctly.

In order to gauge the effectiveness of the RBFNN a naïve Bayes classifier (NB) was employed as a baseline system upon which to improve. Naïve Bayes has been proven to be well suited for text classification throughout literature (Khan *et al.*, 2010; Chang & Huang, 2008; Goller *et al.* 2000; McCallum & Nigam, 1998). One of the main advantages is that naïve Bayes executes linearly, which makes it fast, and it also gives good results, even with little training data available. Figure 1 presents the resulting accuracies of the experiment for each round of cross-validation with an added feature for each iteration.

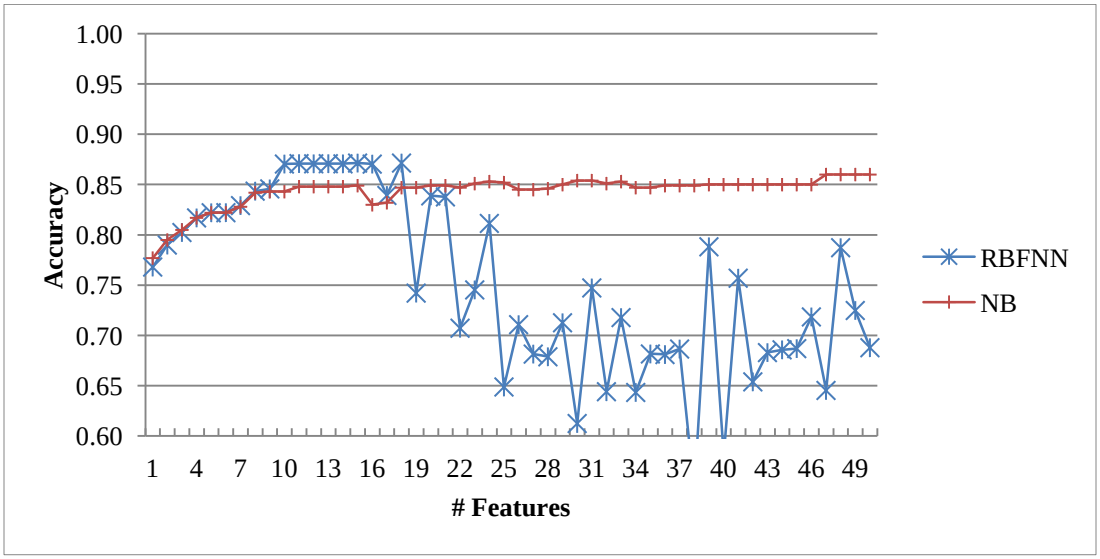

Fig. 1. Accuracy change per added feature

Figure 1 shows a small, but growing trend in accuracy for NB with each added feature up to 50 features. The RBFNN on the other hand shows a steep increase in accuracy for 17-18 extra added features. Thereafter erratic differences in results are noted for each additional feature, resulting in a definitive downward trend. Just before 17-18 extra added features, both algorithms reach a plateau. Within the range of the plateau NB reaches an accuracy high of 84.9% at 15 extra added features. For this same interval the RBFNN reaches an accuracy high of 87.1% (which is also the highest accuracy noted across all added features). The RBFNN outperforms the NB for the particular interval as well as the highest noted NB accuracy of 85.6%. Table 1 summarises the final resulting accuracies for the experiments as follows:

Table 1. Highest noted accuracy per algorithm

Technique	*#Features*	*Accuracy*
NB	15	84.9%
RBFNN	15	87.1%

Encouraging results are noted for both algorithms and the RBFNN shows the best results when the number of features used for training is limited. The steady growth noted for NB is indicative of a robust classifier, which is one of the reasons NB is commonly used as a baseline to compare other classifiers. It is however noted that adding extra features beyond a certain point have little impact on the accuracy of NB. This could be positive when features that are less informative are added to the training data, but negative when informative features do not have as much of an impact.

The erratic results noted for the RBFNN show that extra features have a big impact on the classifier's performance. In this case we see a negative trend when adding extra features indicating that the RBFNN is unable to handle many features and reaches satiety with fewer features.

Conclusion and future work

This study aimed to present an intelligent approach to identify whether a website will be safe to use, or should be avoided due to possible phishing activities. The classification of a site as being malicious or benign was performed by using a Radial Basis Function Neural Network (RBFNN). A bag of words approach to feature extraction was employed on the URL pointing towards the site in question to construct a training data set that was used to train the RBFNN. The RBFNN architecture was determined by means of the BH-RBFNN algorithm that simulates a trial-and-error approach to discern the best number of hidden nodes for the network. A series of cross-validation experiments were performed in order to ascertain the efficiency of the RBFNN algorithm. The results obtained from the experiments for the RBFNN algorithm was compared to a baseline system (based on a naïve Bayes classifier).

The RBFNN was found to exceed the maximum accuracy set by the baseline system. This indicates the suitability of the architecture of the RBFNN that was proposed by BH-RBFNN, for the particular classification problem. It is however noted that the accuracy of RBFNN is found to decrease when the algorithm is presented with too many features. The sensitivity of RBFNN to the quality of features added to its training data is also highlighted. A single extra feature can have an adverse impact on RBFNN performance, but can likewise have a beneficial impact if enough relevant information is conveyed by the feature, given the constraint on the number of features. In order to gain a more representative result the experiments can be repeated on other datasets.

Future work that can be done in order to extend this study includes an investigation into other features (e.g. Term Frequency/Inverse Document Frequency (TF-IDF), Text statistics); investigate other feature selection approaches (e.g. Information Gain); and, including only the most informative features as part of the training data.

References

Anti-Phishing Working Group (APWG). (2014). Phishing activity trends report, 1st Quarter 2014. (Online) http://docs.apwg.org/reports/apwg_trends_report_q1_2014.pdf. Date Accessed: 2015-02-05.

Berger, U. (2012). CSI/FBI Computer Crime and Security Survey 2011-2012. CSI Computer Security Institute.

Broomhead, D. S., & Lowe, D. (1988). Radial basis functions, multi-variable functional interpolation and adaptive networks (No. RSRE-MEMO-4148). Royal Signals and Radar Establishment Malvern (United Kingdom).

Chang, Y. H., & Huang, H. Y. (2008). An automatic document classifier system based on naïve Bayes classifier and ontology. In Machine Learning and Cybernetics, 2008 IEEE International Conference on (6), 3144-3149.

Cranor, L. F. (2008). A Framework for Reasoning About the Human in the Loop. UPSEC, (8), 1-15.

Dhamija, R., Tygar, J. D., & Hearst, M. (2006). Why phishing works. In Proceedings of the SIGCHI conference on Human Factors in computing systems, 581-590. ACM.

Du Toit J.V., Hamadneh N. (2013) Automated Architecture Selection for Radial Basis Function Neural Networks. Technical report: FABWI-N-RKW-2013-365. North-West University, South Africa.

Egan, S. P. (2014). A Framework for High Speed Lexical Classification of Malicious URLs (Doctoral dissertation, Rhodes University).

Finn, A., & Kushmerick, N. (2006). Learning to classify documents according to genre. Journal of the American Society for Information Science and Technology, 57(11), 1506-1518.

Goller, C., Löning, J., Will, T., & Wolff, W. (2000). Automatic Document Classification-A thorough Evaluation of various Methods. ISI, 2000, 145-162.

Jakobsson, M. (2007). The human factor in phishing. Privacy & Security of Consumer Information, 7(1), 1-19.

Baharudin, B., Lee, L. H., & Khan, K. (2010). A review of machine learning algorithms for text-documents classification. Journal of advances in information technology, 1(1), 4-20.

Manjunatha, R., Narayana, P. B., Reddy, K. H. C., & Reddy, K. V. K. (2012). Radial basis function neural networks in prediction and modeling of diesel engine emissions operated for biodiesel blends under varying operating conditions. Indian Journal of Science and Technology, 5(3), 2307-2312.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. In AAAI-98 workshop on learning for text categorization (752), 41-48.

Richardson, R. (2008). CSI computer crime and security survey. Computer Security Institute, 1, 1-30.

Schwarz, G. (1978). Estimating the dimension of a model. The Annals of Statistics, 6(2), 461-464.

Setiono, R. (2001). Feedforward neural network construction using cross-validation. Neural Computation, 13(12), 2865-2877.

Shen, W., Guo, X., Wu, C., & Wu, D. (2011). Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. Knowledge-Based Systems, 24(3), 378-385.