A Danger Feature Based Negative Selection Algorithm

Pengtao Zhang and Ying Tan

Key Laboratory of Machine Perception (MOE), Peking University
Department of Machine Intelligence,
School of Electronics Engineering and Computer Science,
Peking University, Beijing, 100871, China
ytan@pku.edu.cn

Abstract. This paper proposes a danger feature based negative selection algorithm (DFNSA). The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent, laying a good foundation for measuring the danger of a sample. In order to incorporate the DFNSA into the procedure of malware detection, a DFNSA-based malware detection (DFNSA-MD) model is proposed. It maps a sample into the whole danger feature space by using the DFNSA. The danger of a sample is measured precisely in this way and used to classify the sample. Eight groups of experiments on three public malware datasets are exploited to evaluate the effectiveness of the proposed DFNSA-MD model using cross validation. Comprehensive experimental results suggest that the DFNSA is able to reserve as much information of danger features as possible, and the DFNSA-MD model is effective to detect unseen malware. It outperforms the traditional negative selection algorithm based and the negative selection algorithm with penalty factor based malware detection models in all the experiments for about 5.34% and 0.67% on average, respectively.

Keywords: danger feature, negative selection algorithm, feature extraction, malware detection, artificial immune system.

1 Introduction

With the development of immunology, more and more immune mechanisms have begun to be applied in computer security. Forrest et al. first proposed a negative selection algorithm (NSA) to detect the abnormal modification on protected data [1] and later to monitor the UNIX process [2]. Furthermore, they proposed some design principles for computer immune system, such as anomaly detection, diversity and adaptability [3].

The traditional NSA (TNSA) generates a detecting feature library, in which any feature does not match any self, by deleting all the features matching self. It assumes that all the self are harmless and all the non-self are harmful. However, some self are harmful, for example, cancer cells, and some non-self are harmless, taking food as an example.

In order to overcome the drawback of the TNSA in defining the harmfulness of self and non-self, the danger theory (DT) was proposed [4]. According to the DT, the immune system reacts to danger, instead of reacting to non-self, and the internal conversation between tissues and the cells of the immune system controls immunity. The DT explains the autoimmune reaction perfectly.

Based on the DT, Aickelin et al. proposed the danger zone to translate the DT into the field of computer security [5]. From then on, many artificial immune models are proposed, for details, please refer to [6] [7] [8] [9] [10].

Pengtao Zhang et al. proposed a negative selection algorithm with penalty factor (NSAPF) based malware detection (NSAPF-MD) model [11]. The detecting feature library of this model consists of all the non-self danger features, where the features matching self are punished using a penalty factor. It performed well in their experiments. However, this model needs to select a proper penalty factor. What is more, it merely takes advantage of non-self danger features, instead of all the danger features extracted in a training set.

In this paper, a danger feature based negative selection algorithm (DFNSA) is proposed, which reserves the information of danger features to the utmost extent, laying a good foundation for measuring the danger of a sample. On this basis, a DFNSA-based malware detection (DFNSA-MD) model is proposed. It makes use of all the danger features and maps a sample into the whole danger feature space by using the DFNSA. In this way, the proposed DFNSA-MD model measures the danger of a sample precisely and archives good performance.

The remainder of the paper is organized as follows. In Section 2, we introduce the DFNSA. In Section 3, the DFNSA-MD model is presented in detail. Section 4 gives the detailed experimental setup and results. Finally, we conclude the paper with a detailed discussion.

2 Danger Feature Based Negative Selection Algorithm

2.1 Danger Feature

Definition: A danger feature is a feature with dangerous properties, which are able to identify its corresponding dangerous operations. It is the basic element for an immune system to decide whether an immune response should be produced.

In the malware detection field, a danger feature is a code segment which executes a dangerous operation, such as formatting diskette, self-replicating.

There are many expressions for a danger feature. For example, we could use binary string, sequences of assembly codes to express a danger feature in the malware detection field. Generally speaking, a danger feature could appear in both non-self and self. It is the foundation of measuring the danger of a sample.

The danger features can be classified into four categories: (1) danger features only appearing in non-self; (2) danger features appearing in both non-self and self, but tending to appear in non-self; (3) danger features appearing in both non-self and self, but tending to appear in self; (4) danger features merely appearing in self.

2.2 DFNSA

The flow chart of the DFNSA is shown in Fig. 1, where the NCDFL denotes the non-self candidate danger feature library, which is taken as non-self, and the SCDFL means the self candidate danger feature library, which is self.

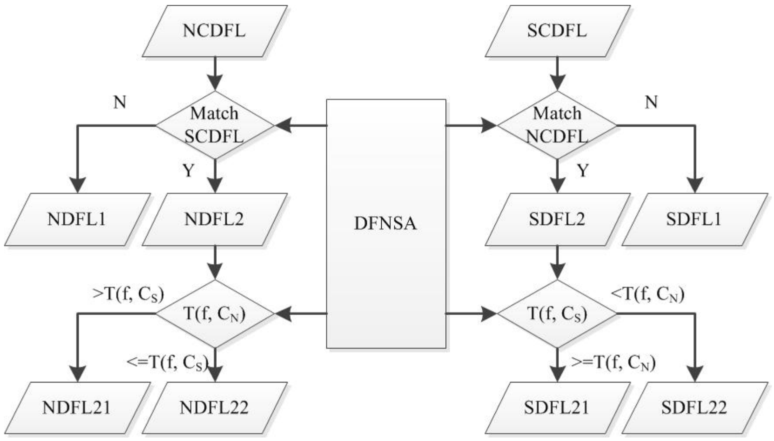

Fig. 1. The flow chart of the DFNSA

Based on the matching of non-self and self, the DFNSA splits the non-self features, which do not match any self, into the non-self danger feature library 1 (NDFL1), and the other non-self features, which match self, into the NDFL2. According to the class tendency of danger features, the NDFL2 is further divided into the NDFL21 and NDFL22, in which the features tend to appear in non-self and self, respectively. The features in the NDFL22 are extracted from non-self, but tend to appear in self, so they are considered to be invalid and deleted.

The measure of the class tendency of a feature is defined as $T(f, C) = P(f, C)$, where $P(f, C)$ denotes the proportion of feature f appearing in class C. if $T(f, C_N) > T(f, C_S)$, then f is considered to tend to appear in non-self, otherwise self. The C_N and C_S denote the classes of non-self and self, respectively.

In a similar way, the SCDFL is firstly split into the SDFL1 and SDFL2 by the DFNSA. Then the SDFL2 is further divided into the SDFL21 and SDFL22. The SDFL22 is deleted with the same reason as the NDFL22.

Definition: if a danger feature f_1 matches a danger feature f_2, the two features are equivalent to each other, written as $f_1 = f_2$.

According to the above definition, since NDFL21 = SDFL22 and NDFL22 = SDFL21, deleting the NDFL22 and SDFL22 merely deletes redundant information, without losing any information of danger features. The proof about NDFL21 = SDFL22 and NDFL22 = SDFL21 is given below.

Proof. $\because \forall f_S \in SDFL2, \exists f_N \in NDFL2, f_S = f_N$
$\therefore T(f_S, C_S) = T(f_N, C_S) = P(f_S, C_S)$ and $T(f_S, C_N) = T(f_N, C_N) = P(f_N, C_N)$
$\therefore$ if $T(f_S, C_S) >= T(f_S, C_N)$, then $f_S \in SDFL21$ and $f_N \in NDFL22$,
if $T(f_S, C_S) < T(f_S, C_N)$, then $f_S \in SDFL22$ and $f_N \in NDFL21$
$\therefore \forall f_S' \in SDFL21, \exists f_N' \in NDFL22, f_S' = f_N'$
and $\forall f_S' \in SDFL22, \exists f_N' \in NDFL21, f_S' = f_N'$,
Similarly, $\forall f_N' \in NDFL21, \exists f_S' \in SDFL22, f_N' = f_S'$
and $\forall f_N' \in NDFL22, \exists f_S' \in SDFL21, f_N' = f_S'$
$\therefore NDFL21 = SDFL22, NDFL22 = SDFL21$

The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent, laying a good foundation for measuring the danger of a sample. The four categories of danger features are stored in the NDFL1, NDFL21, SDFL21 and SDFL1, respectively.

Comparing to the NSAPF, the DFNSA does not need to optimize a penalty factor, dramatically dropping down the training time of the DFNSA, and takes full advantage of all the danger features extracted in a training set.

3 DFNSA-Based Malware Detection Model

In this paper, a danger feature is defined as a code segment which executes a dangerous operation, and expressed as a binary string. The malware and benign programs are taken as non-self and self, respectively.

3.1 Danger Feature Extraction

Malware Instruction Library. This paper defines an instruction as a binary string of length 2 bytes. The class tendency of an instruction i to malware is measured using Eq. 1. The top $P\%$ instructions with the highest tendency value make up the malware instruction library (MIL).

$$I^i = \frac{I_n^i/I_n}{I_n^i/I_n + I_s^i/I_s}, F^i = \frac{F_n^i/F_n}{F_n^i/F_n + F_s^i/F_s}, T^i = \sqrt{(I^i)^2 + (F^i)^2} \tag{1}$$

where I_n^i and I_s^i denote the instruction frequencies of an instruction i in non-self and self, respectively, and F_n^i and F_s^i are the document frequencies of i in non-self and self. I_n and I_s indicate the number of instructions in the non-self and self, respectively. F_n and F_s are the number of samples in non-self and self. I^i and F^i measure the tendency of i to the non-self in the perspectives of instruction frequency and document frequency. T^i is the tendency of i to the non-self.

Since the instructions in the MIL tend to appear in malware, they are dangerous. If the length of a binary string constructed by these instructions exceeds a threshold R bytes, we believe the binary string contains enough danger information and is a danger feature. All the danger features make up the danger feature space.

NCDFL and SCDFL. On the basis of the MIL, the NCDFL and SCDFL are generated by traversing all the malware and benign programs in a training set, respectively. The way to traverse a sample is described below.

A sliding window of length 2 bytes is used to traverse a sample to extract candidate danger features. It moves forward 1 byte at a time. When the window encounters an instruction contained in the MIL, it begins to generate a feature. If the instructions in two adjacent windows do not belong to the MIL, the current feature is terminated as the next feature would not connect with it. If the length of the current feature exceeds R bytes, it is taken as a candidate danger feature. The sliding window keeps on moving to the end of the sample.

This paper sets $R = 4$. The length of a candidate danger feature would be adjusted based on the specific sample and MIL as described above, so R would not affect the result significantly. The frequency of a feature is taken as its weight.

Detecting Feature Library. Taking the NCDFL and SCDFL as the inputs of the DFNSA, four danger feature libraries are generated: NDFL1, NDFL21, SDFL1 and SDFL21, which make up the detecting feature library (DFL) of the proposed DFNSA-MD model. The features in the DFL are the basic elements to construct the danger feature vector of a sample.

3.2 Danger Feature Vector

In this paper, a sample is expressed as a danger feature vector to measure the danger of the sample. The danger feature vector is defined as

$$< \frac{M_{NDFL1}}{L_{NDFL1}}, \frac{M_{NDFL21}}{L_{NDFL21}}, \frac{M_{SDFL1} + M_{SDFL21}}{L_{SDFL1} + L_{SDFL21}} >$$

where M_i denotes the matching value of a sample and a library i, and L_i is the sum of weights of features in a library i, i =NDFL1, NDFL21, SDFL1, SDFL21.

The r-bit continuous matching is taken as the feature matching criteria. Here $r = R * 8$, i.e., the matching part of two features is also a danger feature. The matching value of a sample and a danger feature library is the sum of weights of the features in the library which match any feature of the sample.

The danger feature vector maps a sample into the whole danger feature space, and characterizes a sample efficiently and completely, making the DFNSA-MD model perform well. Every sample in a training set is expressed as a danger feature vector, which is taken as the input of a classifier.

4 Experiments

4.1 Datasets

The experiments in this paper are conducted on three public malware datasets: CILPKU08, Henchiri and VXHeanvens datasets. The three datasets and their composition documents can download from www.cil.pku.edu.cn/resources/.

The benign program dataset used here consists of files of Windows XP and a series of applications, which are the main paunching bag of malware.

Table 1. Experimental platform

CPU	Core 2 Duo 3.00 GHz
RAM	8 GB
Operating system	Win 7 64-bit

4.2 Experimental Setup

The support vector machine (SVM), realized in LibSVM [12], is taken as the classifier of the proposed DFNSA-MD model, and the area under the receiver operating characteristic curve (AUC) is utilized as the performance evaluation criteria. The information of the experimental platform is shown in Table 1.

In the experiments of Section 4.4, eight groups of experiments are taken on the three public malware datasets using 5-fold cross validation, and the 95% confidence intervals are computed to look into the stability of the proposed DFNSA-MD model. Both the CILPKU08 and Henchiri datasets mainly consist of computer viruses, so two experiments are carried on in the two datasets directly, ignoring the categories of malware. The VXHeavens dataset contains 7128 malware which fall into six categories, so we split this dataset into six smaller datasets: backdoor, constructor, miscellaneous, trojan, virus and worm. The miscellaneous includes DoS, Nuker, Hacktool and Flooder, while the malware in the other five smaller datasets, respectively, fall into a category. Six experiments are taken in the six smaller datasets.

In all the experiments, there is no overlap between a training set and a test set. That is to say, to a training set, the malware in a test set are unseen malware. This setting increases the reliability of the experiments.

The TNSA-based malware detection (TNSA-MD) model and the NSAPF-based malware detection (NSAPF-MD) model are imported for comparison.

4.3 Selection of Parameters

This section selects the instruction proportion: $P\%$ used in the MIL, using liner search, where $P = 0.5$, 1.0, ..., 10.0. We do not try larger P, since when $P = 10$, the MIL contains 6553 instructions and already covers a huge danger feature space. The experimental results are shown in Fig. 2.

Fig. 2 illustrates that, with the growth of P, the performance of the DFNSA-MD model shows steady downward trend as the MIL contains more and more instructions with unremarkable tendencies to malware. When $P = 1$, the DFNSA-MD model obtains the optimal AUC = 0.9039.

Generally speaking, the instruction proportion $P\%$ varies with different datasets. Hence we just set the optimization interval of P as [0.5, 3] in the rest of experiments, instead of setting $P = 1$. In the rest of experiments, the P, which makes the DFNSA-MD model perform best in a training set, is set as the optimal P.

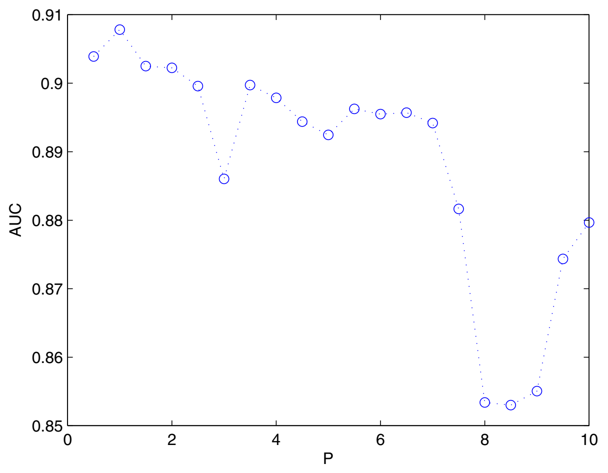

Fig. 2. The experimental results of the selection of parameters

Table 2. Experimental results

	TNSA-MD model	NSAPF-MD model	DFNSA-MD model
CILPKU08	0.9684 ± 0.00568	0.9688 ± 0.00907	0.9761 ± 0.00781
Hechiri	0.9634 ± 0.00755	0.9679 ± 0.01404	0.9808 ± 0.00428
Backdoor	0.8100 ± 0.02060	0.8190 ± 0.01764	0.8247 ± 0.01024
Constructor	0.9095 ± 0.03120	0.9202 ± 0.01545	0.9244 ± 0.01213
Miscellaneous	0.8243 ± 0.01603	0.8255 ± 0.01912	0.8394 ± 0.01028
Trojan	0.7901 ± 0.01332	0.8729 ± 0.01897	0.8735 ± 0.01714
Virus	0.6275 ± 0.01738	0.8746 ± 0.01187	0.8774 ± 0.01784
Worm	0.8252 ± 0.03697	0.8430 ± 0.04788	0.8489 ± 0.04101

4.4 Experimental Results

The experimental results of the proposed DFNSA-MD model are listed in Table 2. The experimental results of the TNSA-MD and NSAPF-MD models are also given in Table 2 for comparison.

From Table 2, the NSAPF-MD model is 4.67% better than the TNSA-MD model in all the experiments on average by making advantage of danger features extracted from malware. The detailed analysis will be given in Section 5.

The DFNSA-MD model outperforms the TNSA-MD and NSAPF-MD models for about 5.34% and 0.67% in all the experiments on average, respectively, without any losing in any experiment. The DFNSA-MD model makes use of all the danger features extracted from a training set, regardless of their categories. Hence the DFNSA-MD model is considered to be able to measure the danger of a sample more precisely, and achieves the best performance.

Table 3. The composition of the DFLs of the three models

	Detecting feature library
TNSA-MD model	NDFL1
NSAPF-MD model	NDFL1, NDFL21, NDFL22
DFNSA-MD model	NDFL1, NDFL21, SDFL1, SDFL21

The 95% confidence intervals of the three models are relatively small from Table 2, indicating that the results of these models are very stable and believable.

5 Discussions

5.1 Comparison of Detecting Feature Library

Table 3 lists the composition of the DFLs of the TNSA-MD, NSAPF-MD and DFNSA-MD models. It is easy to see that the DFL of the TNSA-MD model is the smallest DFL, consisting of NDFL1, i.e., the features merely appearing in non-self. Since the TNSA discards lots of danger features which are believed helpful, the performance of the TNSA-MD model is relatively bad.

The DFL of the NSAPF-MD model consists of NDFL1, NDFL21 and NDFL22, i.e., all the danger features appearing in non-self. The NSAPF reserves the non-self danger features which match self danger features by punishing these features, and obtains a larger DFL. Based on this DFL, the NSAPF-MD model detects malware by measuring the danger of a sample, and achieves good results.

The DFNSA-MD model owns the largest DFL which consists of all the danger features extracted from a training set. The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent. It makes the danger feature vector of a sample contain as much information as possible and measure the danger of a sample better. In this way, the DFNSA-MD model outperforms the TNSA-MD and NSAPF-MD models in all the experiments.

5.2 Comparison of Detecting Time

The detecting time of a sample is proportionate to the number of the features in a DFL. We analyze the average detecting time of the three models for a sample in the virus dataset, in which the average size of a sample is 104 KB.

– The DFL of the TNSA-MD model is the smallest DFL, so it is faster than the other two models to detect a sample, just assuming 0.05 seconds on average.
– The size of the DFL of the NSAPF-MD model lays between that of the TNSA-MD and DFNSA-MD models, taking 0.12 seconds on average for detecting a sample.
– The DFNSA-MD model has the largest DFL, which consists of all the danger features extracted in a training set, so its detecting time is the longest, 0.15 seconds on average, basically meeting the demand of a real-time system.

6 Conclusions

In this paper, the DFNSA has been proposed and applied to detect malware. The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent. Comprehensive experimental results suggest that the DFNSA is able to reserve as much information of danger features as possible, and the DFNSA-MD model is effective to detect unseen malware by measuring the danger of a sample precisely. It outperforms the TNSA-MD and NSAPF-MD models for about 5.36% and 0.67%, respectively.

In future work, we want to find a better way to measure the danger of a sample by importing the danger theory and text categorization methods.

Acknowlegements. This work is supported by the National Natural Science Foundation of China under grants No. 61170057 and 60875080.

References

1. Forrest, S., Perelson, A.S., Allen, L., Rajesh, C.: Self-nonself discrimination in a computer. In: IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, pp. 202–212 (1994)
2. Forrest, S., Hofmeyr, S.A., Somayaji, A., Longstaff, T.A.: A sense of self for Unix processes. In: IEEE Symposium on Security and Privacy, Oakland, pp. 120–128 (1996)
3. Somayaji, A., Hofmeyer, S., Forrest, S.: Principle of a computer immune system. In: New Security Paradigms Workshop, Cumbria, pp. 75–82 (1998)
4. Matzinger, P.: The danger model: a renewed sense of self. Science's STKE 296(5566), 301–305 (2002)
5. Aickelin, U., Bentley, P., Cayzer, S., Kim, J., McLeod, J.: Danger Theory: The Link between AIS and IDS? In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 147–155. Springer, Heidelberg (2003)
6. Ji, Z., Dasgupta, D.: Real-Valued Negative Selection Algorithm with Variable-Sized Detectors. In: Deb, K., et al. (eds.) GECCO 2004, Part I. LNCS, vol. 3102, pp. 287–298. Springer, Heidelberg (2004)
7. Li, Z., Liang, Y.W., Wu, Z.J., Tan, C.Y.: Immunity based virus detection with process call arguments and user feedback. In: Bio-Inspired Models of Network, Information and Computing Systems, Budapest, pp. 57–64 (2007)
8. Li, T.: Dynamic detection for computer virus based on immune system. Sci. China Inf. Sci. 39(4), 422–430 (2009) (in Chinese)
9. Wang, W., Zhang, P.T., Tan, Y., He, X.G.: A hierarchical artificial immune model for virus detection. In: International Conference on Computational Intelligence and Security, Beijing, pp. 1–5 (2009)
10. Wang, W., Zhang, P., Tan, Y.: An Immune Concentration Based Virus Detection Approach Using Particle Swarm Optimization. In: Tan, Y., Shi, Y., Tan, K.C. (eds.) ICSI 2010. LNCS, vol. 6145, pp. 347–354. Springer, Heidelberg (2010)
11. Zhang, P.T., Wang, W., Tan, Y.: A malware detection model based on a negative selection algorithm with penalty factor. Sci. China Inf. Sci. 53(12), 2461–2471 (2010)
12. LibSVM, `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`