ISSN 2250-0987
*Trong-Nguyen Nguyen* et al, UNIASCIT, Vol 4 (1), 2014, 436-440

# FAKE-FOLDER EXECUTABLE FILE RECOGNITION USING ARTIFICIAL NEURAL NETWORK

**Trong-Nguyen Nguyen[#1], Huu-Hung Huynh[#2], Duc-Thuan Huynh[*3]**

[#]*DATIC, Department of Computer Science*
*University of Science and Technology, Danang, Vietnam*
[*] *Government Office, Danang, Vietnam*

*Abstract*—**Computer viruses appear and infect more and more. Humans are still the weakest link in the security chains, so some viruses cheat users by masquerading folder. Most of the current antivirus programs detect viruses based on identifying specific code or behaviours. This paper introduces a new method, which identifies fake-folder executable file using Artificial Neural Network technology. The proposed solution is tested with high accuracy (more than 99.8%).**

***Keywords*— fake-folder, fake-icon, features vector, back-propagation, thresholding**

## I. INTRODUCTION

Since the first days of appearance of early malwares, there is a big contest between virus creators and antivirus experts, and it is becoming more complicated every day, and will continue afterward. At the same time, as antivirus softwares are advancing their methods, on the other hand, the virus writers are seeking for new tactics to overcome them.

Fake-icon viruses are the dangerous malwares that use document symbols (icons) of the popular programs such as Microsoft Word (.doc file), Microsoft Excel (.xls file), or folder to cheat users. The virus is easily fool users execute them. Along with file sharing via USB and other media, fake-icon virus is the malware that has most infected rate today. For example, when the FakeWord viruses infect computer, the entire Word document files of the user will be destroyed or hidden, and replaced by the executable files that have same icon and name of the old documents to trick users. By default, Windows does not show the file extension. Users are at risk of losing information such as passwords or the credit card account. Furthermore, users may lose all storing word documents.

Fake-folder virus is a common form of fake-icon virus type. When the computer is infected, some copies of virus will be created with the icon of the folder, named exist folders or files, and the real folders and files will be hidden. This paper introduces a method, which identifies fake-folder executable files, using Artificial Neural Network (ANN).

ANN is an information-processing model simulates the method of biological neural systems. ANN is widely applied in the field of computer science, specifically the classification problem as: predicting [3] [5] [6], identity [2] [4] [7] [8]… ANN is made from a number of elements (neurons) interconnected via weighted links. An ANN is configured for a particular application through the process of learning from the training set of given samples.

In recognition problem, neural network proved it is more advantageous than other methods (comparing file's icon with an icon template, defining the shape contour...) in that cost very little time for pre-processing procedure, extracting features. In addition, this method is virtually unaffected by noise on the icon image. On the other hand, decision-making methods usually use static settings in the program, so when you want to add new learning models, you need to design the program again. While using the neural network, you simply provide a new training sample, and this is added to "network memory" to calculate new values without affecting the original program structure.

This paper presents an overview of neural networks and application of a back-propagation Multi-Layer Perceptron neural network for building a program which can detect fake-folder executable files. At first, we give a short description on fake-folder viruses and how they work. After that, we present our proposed system with steps and chosen features that need for training and recognition. In the next section, an overview of neural networks will be shown, and we present how to apply a back-propagation Multi-Layer Perceptron neural network to build a program which can detect fake-folder executable files. In the last section, we present a comparison table that shows related detection methods.

## II. RELATED WORK

At this time, there is no specific research on this issue. Some people have experimented with some simple methods such as: compare file's icon with the reference sample icon, determine the shape of the edge.

They scan through the icon image matrix and find the first point has different color with a specified background, and they believe that is the top-left point of the main area in the icon image. From there, the icon image is scanned in parallel with the reference icon image, and the total color difference between two

images is calculated. However, this method has high computational cost and is easy to be affected by noise.

Another idea is scanning an area which has the plus-sign shape in the middle of the icon image, then the authors check each pixel color of the scanned area be yellow or not. After that, they determine edges of the biggest area in the icon image, and then they calculate the bottom-left angle and use it to detect. Disadvantages of this method are that this approach is not really general and brings a low accuracy.

Summary, the above methods are usually affected to noise and have high computational cost; therefore, the rate of accurate detection is very low. Therefore, it is necessary to have a better method to fix those problems. So our approach is almost the first research on this issue.

## III. Proposed Approach

In this section, we present our method for recognizing fake-folder executable file step by step. Proposed solution is built with the following steps: (1) extract icon image of each sample file, (2) calculate each features vector based on icon image, (3) add each vector to the input file, (4) train the neural network with the input file, (5) check files we need and conclude using trained neural network.

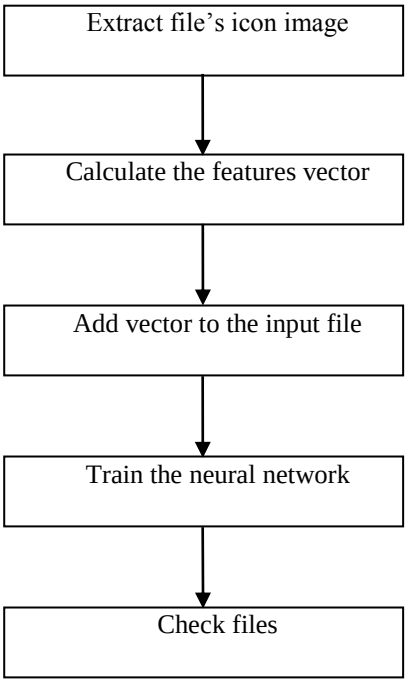


Fig. 1 Processing steps

### A. Input data

Input data are the application's icons that are obtained from the applications, common file format and fake-folder virus samples collected in the research process.

Original data are divided into two data sets:

- Popular icons in Windows operation system
- Icons of fake-folder virus samples collected

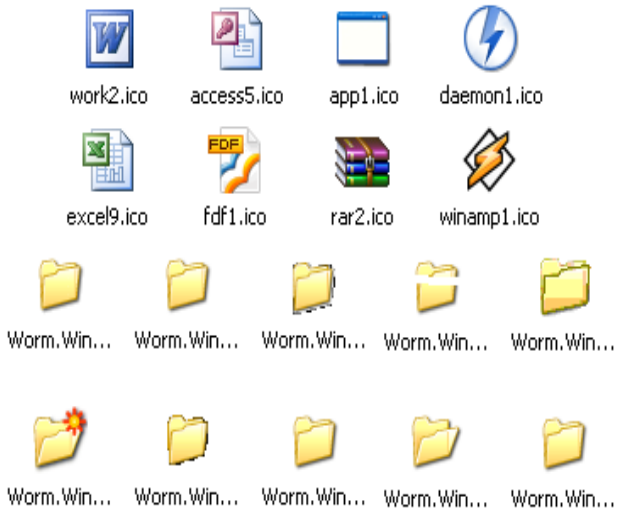


Fig. 2 Some icons in two data sets

### B. Recognition steps

#### 1) Extract each sample file's icon image

We extract the file's icon and convert to bitmap image with three-color channels (RGB).

#### 2) Compute features vector

During doing this research, we found that when we convert the folder image to binary image, only contour (border) of the main area is retained, so the number of black pixels on the binary image is very small when compared with other icons. Therefore, we decided to choose it as an element of features vector to distinguish between folder icon and others. The average values of 3-channels RGB color of original folder image and gray level of gray image are used to distinguish typical "yellow" of the folder with other icons. So the features vector includes five components:

Average color channels Red, Green, and Blue of each pixel on the icon image:

$$v_1 = \frac{\sum_{i=1}^{n} pixel_i.Red}{n} \quad (1)$$

$$v_2 = \frac{\sum_{i=1}^{n} pixel_i.Green}{n} \quad (2)$$

$$v_3 = \frac{\sum_{i=1}^{n} pixel_i.Blue}{n} \quad (3)$$

Average gray level of each pixel on the gray icon image:

$$v_4 = \frac{\sum_{i=1}^{n} pixel_i.GrayLevel}{n} \quad (4)$$

We convert the icon image to binary image using Otsu thresholding method [9]. Then we calculate rate of black pixels:

$$v_5 = \frac{\sum pixel_{black}}{n} \qquad (5)$$

where $n$ is the number of total pixel on icon image.

By combining five above features, we obtain the features vector ($v_1$, $v_2$, $v_3$, $v_4$, $v_5$).

According to the theory of neural network, the result of network training does not depend entirely on each element value of features vector, which depends mainly on the ratio between each element in same dimension of the corresponding feature vectors. Therefore, to speed up calculations, we divide the values $v_1$, $v_2$, $v_3$, $v_4$ by 100, so each element of the vector has a small value.

*3) Add features vectors to the input file*

Each vector is added in the sample vector file line by line; the end of each line is 0 or 1, corresponding to the two sets of samples need to be learned. To improve the training efficiency, we arranged alternating feature vectors of two data sets base on the ratio of the element number. The algorithm is presented below, with C# language:

```
List<string>     Merge2List
                 (List<string> llist,
                  List<string> slist)
{
  List<string> ret =  new
List<string>
         (llist.Count + slist.Count);
  int t = llist.Count % slist.Count;
  int x = 0, j = 0;
  for (int i = 0; i < slist.Count;
i++)
  {
    ret.Add(slist[i]);
    if (x < t)
    {
      ret.Add(llist[x +
  (llist.Count / slist.Count) * x]);
      x++;
      j++;
    }
    for (int k = 0;
 k < llist.Count / slist.Count; k++)
    {
      ret.Add(llist[j++]);
    }
  }
  return ret;
}
```

Each vector is considered as one `string`, each vector set is a `List<string>`. We have two vector sets, with the set has larger element number is `llist`, and the smaller is `slist`. All vectors will be arranged alternating in a new set named `ret`.

*4) Train the artificial neural network*

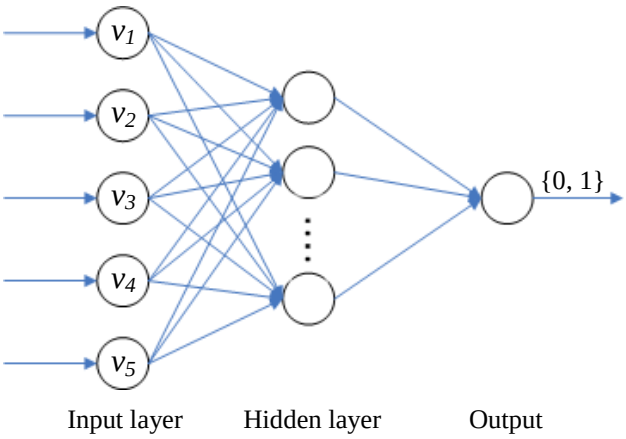


Fig. 3 Our Multi-Layer Perceptron neural network

Currently, the common types of neural networks include: Feed-forward, Feed-back, Self-Organizing. Feed-forward neural network includes many layers of nonlinear processing units. An input vector will be put into the input layer, and then the calculations are spread directly from this layer to the hidden layer and output layer ends. Multi-layer Perceptron (MLP) [1] is one of the typical feed-forward neural networks, which is widely used in recognition systems such as optical character recognition, handwriting, voice recognition.

Because the network has two hidden layers can represent functions with arbitrary shape, so in theory, there is no reason to use the network with more than two hidden layers. It has determined that for the majority of the specific problems, using only one hidden layer network is sufficient. The problems using two hidden layers are rare in practice. Even for the problems need to use more than one hidden layer, then in most cases in practice, using only one hidden layer will give better performance than using more layer. The network training is often very slow when the number of hidden layers to use as much. So our neural network has only one hidden layer.

We designed a MLP network that has three layers: input layer has 5 neurons corresponding to 5 specific elements of the input vector; hidden layer has $n$ neurons (determined by trial and error) and output layer has 1 neuron corresponding output value is 1 (true) or 0 (false) based on approximately the monopole sigmoid function result. Note that we should choose the number of hidden neurons depend the size of training set. The growing of the number of hidden layer neurons will

classify data more accurate, but also increases the computational cost. Network trained by error back-propagation algorithm [2].

*5) Recognizing based on file's icon*

We compute the features vector of each file's icon image and put it to the trained network. If the output value is approximately 1, the icon of this file is the same folder, otherwise.

## IV. Testing Results

Our program is implemented in C# language. We tested the network with training set of 15 fake-folder virus samples and 160 common file type's icons, on a virtual machine 2.01GHz dual-core CPU, 512MB of RAM. The number of features vectors is 175, learning rate is 0.5 and the slope of sigmoid function is 6. The MLP model has 3 hidden neurons. Testing results are shown in the Table 1.

TABLE I
Testing Results

| Scanned | Detected | Wrong detection | Positive rate |
|---|---|---|---|
| 1607 | 72 | 2 | 97% |
| 1607 | 71 | 1 | 98% |
| 1607 | 70 | 0 | 100% |
| 1607 | 70 | 0 | 100% |
| 1607 | 72 | 2 | 97% |
| 1607 | 70 | 0 | 100% |
| 1607 | 70 | 0 | 100% |
| 1607 | 71 | 1 | 98% |
| 1607 | 70 | 0 | 100% |
| 1607 | 70 | 0 | 100% |

We have tested with 70 different fake-folder virus samples. In 10 times of experiments, no fake-folder virus file is missing; results are very positive. Some files are false detected (a very low rate; this rate is acceptable) because the original sample set is not rich enough, and the training results differ after each training.

After the network is trained from the sample icons (folder icons of the Windows XP), our program can identify the more complex folder icon, and discover all types of the Windows 7 folders (several those icons are shown in Fig. 4).

Fig. 4 Some detected icons on Windows 7 (with noise)

Fig. 5 shows several icons are usually detected as folder icon, but they are not similar to folder icon. The wrong detection happens because in those icons, the "yellow" color is mainly, and the numbers of black pixels on the binary icon image are rather small. So after a few network training, the received weight matrix will identify those icons are the same icon as a folder.

Fig. 5 Several icons are detected false

There are many ways to select features for an icon image. In this paper, we only give five features because they are simple, easy to understand and install. And most importantly, they are features that clearly distinguish the difference between the folder icon and other icons.

In addition to the above features, we can test a number of additional features such as perimeter, area, line-scan or chain code of boundary, the radial distance of the icon image.

## V. Conclusion and Discussion

In this paper, a new robust method based on artificial neural network is proposed for detecting the fake-folder executable files. With the results obtained during testing, our network performances fairly stable with set parameters. The result is hardly influenced when the icon image has noises. After training, the recognition has high accuracy, all files have icons that are the really same folder icons (of Windows XP and Windows 7) are detected, identification speed is relatively fast (less than 30 seconds to check more than 1600 files), more than 99.8% accuracy. Our method has proven very clearly the advantages of it for identification. The main advantage of this system is almost unaffected by noise, cost very little time for pre-processing procedure, extracting the features.

After each training session, a new weighted matrix will be created, so the test results will be difference. Therefore the network is needed to train and re-test several times to choose the appropriate weight matrix. The main advantages of this method are presented in the Table 2.

TABLE II
Methods Comparison

| | Our approach | Previous methods |
|---|---|---|
| Affected by noise | No | Yes |
| Generality | Yes | No |
| Computational cost | Low | High |

To obtain the recognition results with higher accuracy, it is necessary to obtain larger number of icon samples (especially icons of common file types on the Windows operating system). In addition, we need to adjust the number of neurons in hidden layer and the network parameters accordingly, and it requires time and network training test longer.

We can extend the recognition capabilities of the network for other fake-icon viruses by providing training samples corresponding to the network. Our

method is particularly useful when combined with other antivirus products, in the current context a large proportion of viruses is fake-folder. In further work, we will try to recognize these virus types base on their shapes using some edge descriptors.

## ACKNOWLEDGMENT

This work was supported by the DATIC, Department of Computer Science, University of Science and Technology (DUT), The University of Danang, Vietnam.

## REFERENCES

[1] Dave Anderson, George McNeill, *Artificial Neural Networks Technology*, Prepared for Rome Laboratory RL/C3C Griffiss AFB, NY 13441-5700, USA, 2006.

[2] Ngo Xuan Bach, "Ung dung mang neuron trong nhan dang chu viet tay roi rac han che truc tuyen tren Tablet PC", B.S. thesis, University of Engineering and Technology, Vietnam Nation University, Hanoi, Vietnam, 2006.

[3] Doan An Hoa, Le Quoc Nam, "Ung dung mang neuron xay dung he thong phat hien xam nhap", B.S. thesis, Danang University of Technology, Danang, Vietnam, 2011.

[4] Do Thi Phu, "Nhan dang ky tu quang hoc bang mang neuron", B.S. thesis, Danang University of Technology, Danang, Vietnam, 2008.

[5] Paula Odete Fernandes, João Paulo Teixeira, João Matos Ferreira, Susana Garrido Azevedo, "Forecasting Tourism Demand with Artificial Neural Networks", Book of Proceedings Vol.II – *International Conference On Tourism & Management Studies*, 2011, pp. 1017-1019.

[6] Phunsak Theramongkol, "Intelligent Ozone-level Forecasting Using Artificial Neural Network", M.S. thesis, Environmental Systems Engineering University of Regina, Canada, 2000.

[7] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, Bernard Manderick, "Credit Card Fraud Detection Using Bayesian and Neural Networks", Vrije Universiteit Brussel, Brussel, 2002.

[8] Henry A. Rowley, Shumeet Baluja, Takeo Kanade, "Neural Network – Based Face Detection", presented at the *IEEE Computer Vision and Pattern Recognition*, CA, June 18-20, 1996.

[9] Qian Wang et al., "Otsu thresholding segmentation algorithm based on Markov Random Field", *Seventh International Conference on Natural Computation (ICNC)*, Vol. 2, July 2011, pp. 969 - 972.