http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 6 (3): 370-375, 2007
ISSN 1812-5638

Automated Oracle Based on Multi-Weighted Neural Networks for GUI Testing

[1]Mao Ye, [1]Boqin Feng and [2]Li Zhu
[1]School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[2]School of Software, Xi'an Jiaotong University, Xi'an 710049, China

Abstract: Graphical User Interfaces (GUI) software has characteristics different from traditional software. The oracle for GUI software testing must validate the correctness of the GUI. An automated oracle based on multi-weighted Neural Networks (NN) is proposed in this paper to validate the GUI from users' viewpoint. In this approach the multi-weighted NN is used to learn the topological information in the feature space for the expected images of the graphical interface. The topological information is then used to verify the correctness of the GUI. By this method, trivial difference in the graphical interfaces can be ignored and GUI be automatically tested in the manner of human being. Experimental results show the method is of potential application in automated GUI testing.

Key words: Oracle, graphical user interfaces (GUI), software testing, neural networks

INTRODUCTION

Automated oracle includes the capabilities to generate expected output and compare it with actual output automatically (Memon *et al*., 2003). It is one of the most difficult and expensive parts in software testing process. A perfect oracle would be behaviorally equivalent to the Application Under Test (AUT) and completely trusted. It would always produce a correct result for every input specified by software specification. It is therefore as difficult as solving the original design problem to develop a perfect oracle. As a result, very few techniques have been implemented to automate oracle. When testing the traditional software, a tester is assumed to provide the expected behavior of the software in the form of a table of pairs (Peters and Parnas, 1994), logical expressions to be satisfied by the software (Bousquet *et al*., 1999), or temporal constraints that specify conditions that must not be violated during software execution process (Dillon and Ramakrishna, 1996). However, to generate expected outputs for GUI is different from the traditional software because the expected outputs of the GUI software are a lot of images or states of the graphical interfaces. When capture/replay tool is used to test the GUI, expected outputs are saved for comparing while recording test scripts or inserted in the test scripts manually (Ostrand *et al*., 1998; Andersson and Bache, 2004). The comparison process is usually implemented by bit-to-bit verification between the expected and actual images. Chen *et al*. (2005) proposes the concept to integrate specification-based and capture/replay based approaches to test GUI software. Test points are inserted in the test flow. Memon presents planning method to generate expected results for GUI (Memon *et al*., 2005). The oracle is implemented by comparing the actual and expected states of the graphical interfaces. It must construct GUI model and set conditions for every operator manually, which need a lot of time and cost. When testing the GUI by the technique of Finite State Machines (FSM) (Chen and Subramaniam, 2002), a model of GUI must be constructed in the form of FSM. It may result in state explosion problem because the state of the GUI is very large (White *et al*., 2003).

Biomimetic Pattern Recognition (BPR) is proposed by Wang recently (Wang, 2002; Wang and Lai, 2005). It can simulate the function of human being, which is different from the traditional statistical pattern recognition method, such as Support Vector Machines (SVM) (Vapnik, 2000) that uses optimal classification hyperplane as main principle. As a result, it can recognize but not classify objects. We propose the automated oracle model based on BPR theory in this research to test GUI from users' viewpoint. Multi-weighted Neural Networks (NN) are used to implement the oracle. The appeal of the BPR lies in its ability to recognize the image objects in the manner of human being after training. Few experiments have been conducted to validate that the method is effective.

BPR AND MULTI-WEIGHTED NEURAL NETWORKS

Traditional pattern recognition method aims at finding the optimal classification for the different classes of samples in the feature space. However, BPR try to find

Corresponding Author: Mao Ye, School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China

the optimal coverage for the samples in the same class. It uses the empirical knowledge that the samples in the same class are continuous and gradually changed. For any two samples in the class, there must a sequence of samples in the same class, such that these samples change gradually in the feature space R^n (Wang, 2002). Let A be a sample set of a class. For $\forall x, y \in A$ and $\varepsilon>0$, there must be a set B, such that:

$$\begin{aligned} &B = \{x_1, x_2, \cdots, x_n \mid x_1 = x, x_n = y, x_n \in A \\ &x_i \in A, \rho(x_i, x_{i+1}) < \varepsilon, i = 1, \cdots, n-1\} \end{aligned} \tag{1}$$

where ρ is a difference function. The set A can be covered by:

$$P_A = \{x \mid \rho(x, y) \le \theta, y \in A, x, y \in R^n\} \tag{2}$$

where $\theta>0$ is constant which describes the effect of the noise and controls the area of the region covered. The process of determining if a sample belongs to a class is to check if the sample belongs to P_A. As a result, BPR can recognize samples in the style of human being. The region P_A can be implemented by several multi-weights neurons, which construct a NN. A multi-weights neuron can be described as (Wang *et al*., 2003):

$$y = f[\phi(w_1, w_2, \cdots, w_m, x) - \theta] \tag{3}$$

where w_i, i = 1,2,..., m are weight vectors, x is input vector, ϕ is neuron's computation function and f is neuron's activation function. When m is 2, the Eq. (3) became a two-weighted neuron. It has the mathematical description as follows:

$$y = f[\phi(w_1, w_2, x) - \theta] \tag{4}$$

where $\phi(w_1, w_2, x)$ is the distance from point x to the line segment w_1w_2 which is defined as follows:

$$\phi(w_1, w_2, x) = \begin{cases} \text{IF} p(w_1, w_2, x) > 0 \\ \|x - w_1\| \\ \text{IF} p(w_1, w_2, x) < -\|w_1 - w_2\| \\ \|x - w_2\| \\ \text{Otherwise} \\ \sqrt{\|x - w_1\|^2 - p^2(w_1, w_2, x)} \end{cases} \tag{5}$$

where

$$p(w_1, w_2, x) = < (x - w_1), \frac{w_1 - w_2}{\|w_1 - w_2\|} > \tag{6}$$

In Eq. (4), f is an activation function which is defined as follows:

$$y = \begin{cases} 1 & , \phi(w_1, w_2, x) - \theta \le 0 \\ 0 & , \text{otherwise} \end{cases} \tag{7}$$

The region covered by a two-weighted neuron in feature space R^n is:

$$C_i = \{x \mid \phi(w_1, w_2, x) \le \theta, x \in R^n\} \tag{8}$$

where w_1 and w_2 are two sample of a class in R^n. Several neurons construct a NN which can form a complicated region. The region covered by all m neurons is

$$C_A = \bigcup_{i=1}^{m} C_i ,$$

which is used to approximate the region of P_A. If a new sample is in the region of C_A, the output of the NN is 1, which means that the new sample is in the same class of A. Otherwise, the output of the neuron networks is 0, which means the NN reject to recognize it because the difference between the new sample and A is larger than θ. Therefore, BPR can recognize the object from the viewpoint of human being, which can be used to implement the oracle for GUI testing.

ORACLE MODEL BASED ON MULTI-WEIGHTED NEURAL NETWORKS

Automated oracle model for GUI testing: Here an automated oracle model is proposed to test GUI. GUI software is different from the traditional one because it supplies graphical interfaces by which the users operate the GUI software to implement tasks. A failure in GUI may affect the users to use the GUI software. Oracle for GUI testing is to validate the correctness of GUI when software is running. Each graphical interface includes a set of windows such as button and edit box, by which users can input data and get outputs. The state of a GUI at the time t is a set:

$$S = \{(w_i, p_j, v_k) \mid w_i \in W, p_j \in P, v_k \in V\} \tag{9}$$

where W is the set of windows, P is the set of window properties and V is the set of values on window properties. When the software accepted the events from users, it transfers from one state to another. The state S appears as an image (or graphical interface) in front of the users when software is running. Let g_1 and g_2 be two images of a graphical interface, s_1 and s_2 be samples in

feature space R^n of these two images. We define $\varphi(s_1, s_2)$ be difference between s_1 and s_2. If s_1 is exactly same as s_2, $\varphi(s_1, s_2)$ is equal to 0. The larger value of $\varphi(s_1, s_2)$ means the larger difference. Now let g_e and g_a be the expected and actual images for the same graphical interface, s_e and s_a be samples in the feature space R^n of the images g_e and g_a, respectively.

The general model of oracle is in nature to generate the s_e and check if $\varphi(s_a, s_e)$ is equal to 0. If it is, there is no failure. Otherwise, a failure is reported. However, it is not suitable in GUI testing. If there is a little difference between the actual and expected images of the graphical interface when the GUI software is running, it is often not regarded as a failure from the users' viewpoint. For example, the position of a button is several pixels different from the expected position, which can not be taken as failure because it will not affect the usage of the software. Therefore, we propose the new oracle model for GUI testing. Construct a set

$$G = \{x \mid \varphi(x,s) \le k, s \in D, x, s \in R^n\} \quad (10)$$

where D is sample set of the acceptable images, $k \ge 0$ stands for the acceptable difference between the samples of the expected and actual images from the user's viewpoint. Automated oracle is to determine G from the training samples and check If $s_a \in G$. If it is, there is no failure. Otherwise, a failure is reported. One merit of the proposed oracle is that it can keep the key feature of the GUI and at the same time ignore the small difference among expected images and actual ones. As a result, the oracle is trivial difference insensitive and can be used to test GUI from user's viewpoint. The other merit of the oracle is that the parameter k can be used to control the test precision. If a high precision is needed, the values of k can be set to a little number. It will reduce the area of the set G in feature space. As a result it can just accept the images that have small difference from expected images. In fact the function $\varphi(.)$ and parameter k in Eq. (10) in nature have the same meanings as the function $\rho(.)$ and the parameter θ in Eq. (2). As a result, the function $\rho(.)$ and the parameter θ will be used in the next sections.

Oracle based on multi-weighted neural networks: The automated oracle model based on multi-weighted NN is as Fig. 1. Acceptable images of graphical interfaces are preprocessed and transformed into samples in R^n. These samples are used to train the multi-weighted NN. When the training process finishes, trained NN can cover the region of G in the Eq. (10). Trained NN is then used to recognize the actual images of the graphical interface. If the sample s_a of an actual image is in G, there is no failure. Otherwise, a failure is reported. Let $S = \{s_1, ..., s_t\}$ be the training set for a graphical interface and $|S| \ge 2$.

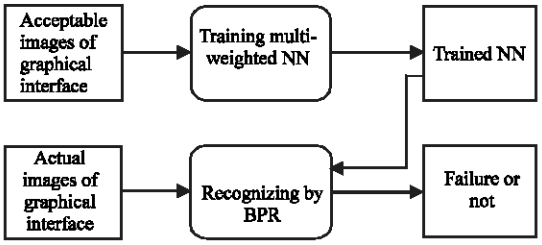


Fig. 1: Automated oracle model for GUI testing

The process of training multi-weighted NN is summarized as follows.

Step 1: Set the value of parameter θ.
Step 2: Select a sample s_i from the set S randomly.
Step 3: Calculate the distance between s_i and the samples in $S-\{s_i\}$. Get the sample $s_j \in S-\{s_i\}$ such that $\rho(s_i, s_j) = \min_{s \in S-\{s_i\}} \rho(s_i, s)$. Tie can be broken randomly.
Step 4: Construct a two-weighted neuron by s_i and s_j which covers the region:

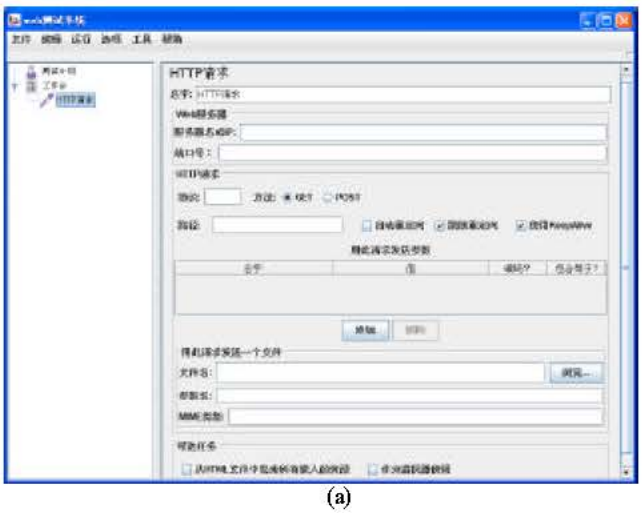
$$C_i = \{x \mid \phi(s_i, s_j, x) \le \theta, x \in R^n\} \quad (11)$$

Step 5: Add the neuron into NN. Let $S \leftarrow S-\{s_i\}$, $s_i \leftarrow s_j$.
Step 6: If $S = \{s_i\}$, the training process finishes. Otherwise, go to step 3.

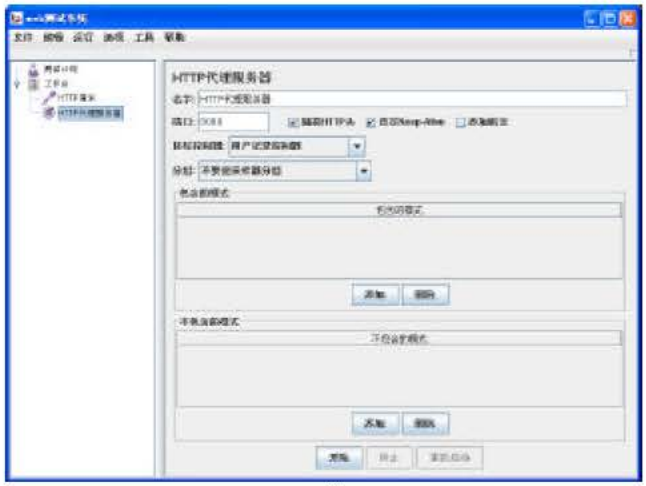
The process of recognizing by BPR is implemented by checking if the sample of the image is in the region covered by trained NN. If it is, the output of the NN is 1 and there is no failure. Otherwise, the output of the NN is 0 and a failure is reported.

EXPERIMENTS

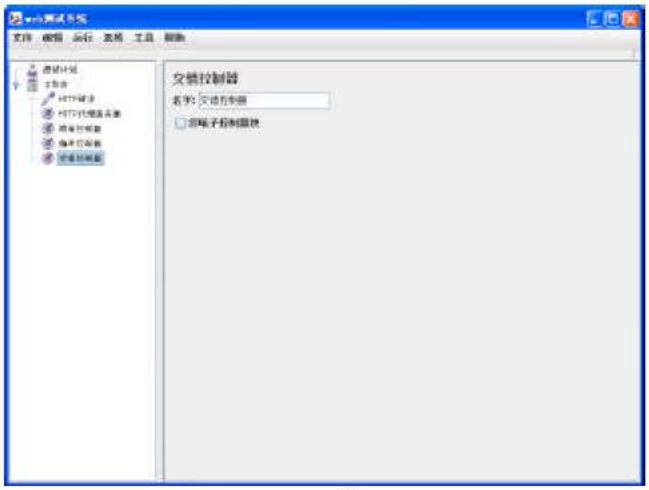
In the experiment, we collect three classes of samples. Each class corresponds to a graphical interface. These three interfaces are named as I_1, I_2 and I_3, respectively. Figure 2 shows an expected image for each interface. For every interface I_j, $j = 1, 2, 3$, we select 5 images that have small difference from each other and can be accepted by users. These 5 images are used to train a NN for the interface I_j. As a result, the number of NN constructed in the experiment is 3 and the number of neurons in each NN is 4. After training process has finished, we collect another 100 images from these 3 interfaces and other 7 interfaces which are named as $I_4, ..., I_{10}$, respectively. Each interface I_j, $j = 1, ..., 10$, supplies 10 images. The 10 images collected from each I_j, $j = 1, 2, 3$, consist of two types. Five samples are acceptable and the other five samples are not



(a)



(b)



(c)

Fig. 2: The expected images of the three graphical interfaces

acceptable from user's viewpoint. These 100 images are used for testing. All images are transferred into gray scale and the resolution is changed into 33×25 to form the samples in feature space R^{825}. The correct acceptance rate is defined as $\eta = \alpha_1/\alpha_2$, where α_1 is the number of test samples recognized correctly by three trained NNs and α_2 is the number of test samples which are acceptable by users for the interfaces I_1, I_2 and I_3. As a result α_2 is equal to 5×3 = 15. The rejection rate is defined as $\gamma = \beta_1/\beta_2$, where β_1 is the number of test samples rejected correctly by three trained NNs and β_2 is the number of test samples which should be rejected. As a result β_2 is equal to 5×3+10×7 = 85.

By the experiments it is concluded that the value of θ is important for the result of GUI testing. The method proposed can verify the GUI automatically and effectively provided that the value of θ is set properly. The value of θ determine the area of the covering region. When the image is in the covering region of the corresponding graphical interface, the output of the NN is 1 and it means there is no failure. Otherwise, the output of the NN is 0 and it exposes a failure because the images can't be recognized by NN. In the one hand, the large value of the parameter θ will make the region covered by NN so large that it will cover the sample of the other class wrongly. In the other hand, the small value of the parameter θ will make the region covered by NN so small that it cannot include all samples in the same class. When setting θ to 21, the correct acceptance rate η is about 96% and the rejection rate γ is about 99% after experimenting for 10 times. The results show when θ is selected properly, the method proposed has very high correct acceptance rate and rejection rate for GUI testing. It means a large part of images that are slightly different from expected images but accepted by users can be recognized correctly. At the same time, the images which have much difference from expected images can be rejected correctly. As a result the oracle based on multi-weighted NN can effectively test GUI in the manner of testers.

CONCLUSIONS

A model of automated oracle based on multi-w eighted NN is proposed in this paper to automatically test the graphical interfaces of GUI software. By the experimental results it can be concluded that the method verifies graphical interfaces effectively. GUI can be tested from the viewpoint of testers by the automated oracle model. It ignores the irrespective difference and compares the key feature of the GUI. The correct acceptance rate and rejection rate are very high, which shows that GUI can be tested effectively. By the method GUI testing can be automated without constructing GUI model and setting conditions for each operator manually. And there is no problem of state explosion which occurs when modeling the GUI software by FSM. By the oracle proposed a lot of time and software testing cost can be saved because the process of verification is implemented automatically.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their careful reading of this paper and for their helpful and constructive comments. This work was supported in part by the National High Technology Development Plan of China (863) under grant no. 2003AA1Z2610.

REFERENCES

Andersson, J. and G. Bache, 2004. The Video Store Revisited Yet Again: Adventures in GUI Acceptance Testing. LNCS 3092, Springer-Verlag Berlin Heidelberg, pp: 1-10.

Bousquet, L.D., F. Ouabdesselam, J.L. Richier and N. Zuanon, 1999. Lutess: A specification-driven testing environment for synchronous software. In Proceedings of the 21th International Conference on Software Engineering.

Chen, J. and S. Subramamiam, 2002. Specification-based testing for GUI-based applications. Software Quality J., 10: 205-224.

Chen, W.K., T.H. Tsai and H.H. Chao, 2005. Integration of specification-based and CR-based approaches for GUI testing. In: Proceedings of the 19th International Conference on Advanced Information Networking and Applications.

Dillon, L.K. and Y.S. Ramakrishna, 1996. Generating oracles from your favorite temporal logic specifications. ACM SIGSOFT Software Engineering Notes, 21: 106-117.

Memon, A., I. Banerjee and A. Nagarajan, 2003. What test oracle should I use for effective GUI testing. In Proceedings of the 18th IEEE International Conf. on Automated Software Engineering.

Memon, A., A. Nagarajan and Q. Xie, 2005. Automating regression testing for evolving GUI software. Journal of Software Maintenance and Evolution: Research and Practice, 17: 27-64.

Ostrand, T., A. Anodide and H. Foste *et al*., 1998. A visual test development environment for GUI systems. ACM SIGSOFT Software Eng. Notes, 23: 82-92.

Peters, D. and D.L. Parnas, 1994. Generating a test oracle from program documentation. In Proceedings of the International Symposium on Software Testing and Analysis.

Vapnik, V.N., 2000. The Nature of Statistical Learning Theory. 2nd Edn. Springer-Verlag. New York.

Wang, S.J., 2002. Bionic (topological) pattern recognition-a new model of pattern recognition theory and its applications. Acta Electronica Sinica, 30: 1417-1420.

Wang, S.J., J. Xu, X.B. Wang and H. Qin, 2003. Multi-camera human-face personal identification system based on the biomimetic pattern recognition. Acta Electronica Sinica, 31: 1-3.

Wang, S.J. and J.L. Lai, 2005. Geometrical learning, descriptive geometry and biomimetic pattern recognition. Neurocomputing, 67: 9-28.

White, L., H. Almezen and S. Sastry, 2003. Firewall regression testing of GUI sequences and their interactions. In Proceedings of the International Conference on Software Maintenance.