User Interface Evaluation with Machine Learning Methods

by

Yanxun Mao

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in the University of Michigan
2019

Doctoral Committee:

Professor Yili Liu, Chair
Professor Daniel Burns
Assistant Professor Clive D'Souza
Assistant Professor Xi Jessie Yang

Yanxun Mao

myx@umich.edu

ORCID iD: 0000-0002-7968-7110

To

Xuedong Mao

Yan Yan

Changyu Yan

Pojing Liu

&

Jimei Yan

Acknowledgements

I would like to reflect on the people who have supported and helped me throughout my PhD study. Firstly, I would like to express my sincere gratitude to my adviser Dr. Yili Liu for the continuous support of my research, for his patience, motivation, and valuable guidance. Equally important appreciation must go to Dr. Daniel Burns, Dr. Clive D'Souza and Dr. Xi Jessie Yang who kindly served as my dissertation committee members and provided insightful comments, tremendous encouragement, and valuable suggestions. Their willingness to give their time so generously has been very much appreciated.

I would also like to extend my thanks to many professors, Paul Green, Seth Guikema and Nadine Sarter, as well as many staff members in the Horace H. Rackham School of Graduate Studies, and Matt Irelan, Wanda Dobberstein, Tina Picano Sroka from Department of Industrial and Operations Engineering at the University of Michigan for providing me fellowship and graduate student instructor positions that support my study and dissertation work.

Finally, I must express my very profound gratitude to my parents and to my girlfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Table of Contents

List of Tables

List of Figures

Abstract

With the increasing complexity of user interfaces and the importance for usability evaluation, efficient methods for evaluating the usability of user interfaces are needed. Through this dissertation research, two computational models built with machine learning methods are introduced to evaluate user interface usability.

This research consists of two phases. Phase I of the research implements the method of support vector machine to evaluate usability from static features of a user interface such as widget layout and dimensions. Phase II of the research implements the method of deep Q network to evaluate usability from dynamic features of a user interface such as interaction performance and task completion time.

Based on the research results, a well-trained Phase I model can distinguish and classify user interfaces with common usability issues and is expected to recognize those issues when sufficient data is provided. Phase II model can simulate human-interface interaction and generate useful interaction performance data as the basis for usability analysis. The two phases of the research aim to overcome the limitations of traditional usability evaluation methods of being time-consuming and expensive, and thus have both practical and scientific values. From the practical perspective, this research aims to help evaluate and design user interfaces of computer-based information systems. For example, today's application software development on computer based information system always integrates many functions or task components into one user interface page. This function integration needs to be carefully evaluated to avoid usability issues

and the competitive field of software development requires an evaluation process with short cycles. Phase I and Phase II of the research provide an efficient but not necessarily comprehensive usability evaluation tool to meet some of the demands of the field. From the scientific perspective, this research aims to help researchers *make quantifiable predictions and evaluations* of user interfaces. Qualitative theories and models are important, but often insufficient *for rigorous understanding and quantitative analysis*. Therefore, this research work on computational model-based interface evaluation has *important theoretical value* in advancing the science of studying human behavior in complex human-machine-environment systems.

Chapter 1 Introduction

Chapter Summary

This chapter first introduces the motivation of this dissertation research. Next, it clarifies the research scope including the user interface type, the interface attributes to evaluate, and their definitions. Then, it briefly discusses two phases of the research including their research goals, methods and relationship between them. Lastly, it discusses the practical and scientific values of the research.

1.1 Research Motivation

User interface as a connection between humans and machines is widely used in almost every field of work. High penetration rates of the computer, Internet and portable devices also verify that interacting with user interface has become part of many people's lives. It will be very beneficial to properly design user interfaces, which could improve efficiency of work, reduce errors or bring convenience to people's lives.

In the design of user interfaces, one problem is the contradiction between the integration of functions or information in user interfaces and the limited information processing resources of humans. Heavily function-integrated user interfaces or information intensive user interfaces could lead to failure of information acquisition, incorrect operations or even some lethal results. Another problem gradually shows up when an increasing amount of applications of portable

devices go into market. Due to the short development cycle of those applications, the time-consuming and high cost limitations of lab-based user interface evaluations are magnified and create difficulty in following the pace of development. These two problems together motivate this research effort.

1.2 Scope of the Research

In the following, types of user interfaces to be studied, attributes of user interfaces to be evaluated, and types of evaluation methods to be built in the research will be discussed.

1.2.1 Types of User Interfaces

Based on different input and output sources, user interfaces can be categorized into physical panel user interface, touch screen user interface, and so on. Different types of user interfaces have different interactive modes. There may exists one general evaluation method that can consider all the features of these interactive modes and evaluate all types of user interfaces with one model. However, it creates too much difficulty for data collection and analysis. Therefore, as a stepping stone for user interface evaluation with machine learning method, this research only focuses on one specific type of user interface, i.e., visual user interface on computer based information systems. To reduce the complexity and difficulty of modeling, user interfaces used for evaluation in this research are restricted to widgets button and textbox. Cursor navigation and left click of mouse are the only two operations to interact with user interface in the domain of this research.

1.2.2 User Interface Attributes to Evaluate

As mentioned in the Research Motivation section, properly designing a user interface has significant values in many aspects. Systematically evaluating a user interface is the foundation of successful design. Two aspects of user interfaces are usually used for evaluation: usability and utility. They reflect two groups of interaction performance between user interfaces and users. This dissertation research focuses on usability evaluation. Usability has the dictionary meaning of ease of use and it is not an exclusive attribute of user interface evaluation. International Organization for Standardization (ISO 9241-11, 1998) defines usability as "*The effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments. effectiveness: the accuracy and completeness with which specified users can achieve specified goals in particular environments.*". Jakob Nielson and Ben Shneiderman (1993) define usability for user interface from five aspects: learnability, efficiency, memorability, errors, and satisfaction. Learnability, efficiency and memorability refer to the ease of accomplishing tasks when the system is used by a novice user, used by an expert user and only used occasionally, respectively. Errors can be counted during performance observation and rated based on severity. Satisfaction refers to how pleasant a user feels when using the design. The main method of this research is to capture user's responses as basis for statistical analysis. Definition of usability in this research needs to be clear and operational. Therefore, usability in this research is defined from two aspects: efficiency and satisfaction.

Efficiency is defined as the ease of accomplishing tasks which can be measured by quantities such as task completion time in this research. In Nielson and Shneiderman's definition, it can be noticed that the performances of novice user, expert user and occasional user are distinguished, because for different groups of users the same user interface may provide

different experience. However, in this research with machine learning method, this difference only appears in the training data of different groups of users and does not generate difference at the methodology level. Once evaluation model for efficiency is built it can generate evaluation results for learnability, efficiency and memorability, as defined by Nielson and Shneiderman, by inserting the training data of different groups of users. Therefore, efficiency is defined without specifying user groups. During experiment, data collected to train the model is only from novice users due to time and budget limitations. Thus, experimental results actually reflect evaluation of learnability in Nielson and Shneiderman's definition.

Satisfaction is defined as how pleasant a user feels when viewing the design. This definition in comparison with Nielson and Shneiderman's definition more emphasizes the user interface's function of presenting information. In other words, satisfaction in this research focuses on evaluating the static features of user interface.

Error is not included in this evaluation for two reasons. First, error in comparison with other aspects of usability is complicated. It is hard to define error in user interface interaction. For example, trivial actions or mis-clicks are difficult to classify. Also, it is hard to determine whether all errors should be counted equally. Second, during data collection error cannot be forced. Studying error in user interface evaluation requires a large amount of data. Therefore, in the domain of the current research, error is not evaluated.

1.2.3 Types of Evaluation Methods

Currently, there exist many types of user interface evaluation methods. These evaluation methods can be divided into three main types as shown in Figure 1-1.

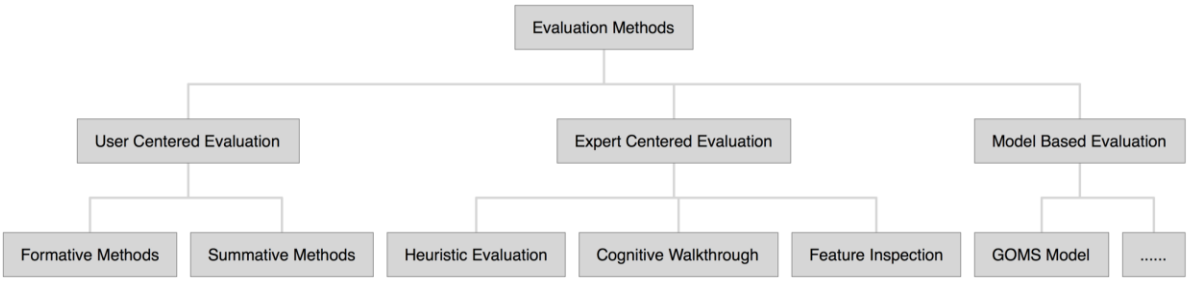


Figure 1-1: Three types of user interface evaluation methods: User centered evaluation, expert centered evaluation, and model based evaluation *(Scholtz, 2004)*

User centered evaluation mainly refers to empirical evaluation with experiments on real users or potential users with methods such as formative methods and summative methods. **Expert centered evaluation** mainly refers to evaluation performed by expert evaluators, including formal evaluation with some analysis technique and heuristic evaluation. **Model based evaluation** mainly refers to evaluation with computerized procedures such as GOMS model. To compare these types of evaluation methods, two aspects are taken into account, method cost and method performance. **Method cost** is used to describe the time, expense and labor cost of the evaluation method. **Method performance** is used to describe the amount of usability issues the method can compare or recognize. User centered evaluation method has the best method performance by being able to discover most usability issues under lab condition, but it is also characterized with have the time consuming and labor intensive limitations. Model based evaluation method has the lowest method cost but limited feedback of usability issues. Expert centered evaluation method is a compromise between method cost and method performance. (Scholtz, 2004)

In Research Motivation, the second problem raises a new demand to reduce the method cost. To optimize lab controlled experiment procedures with the same method performance, it is difficult to reduce the method cost since the cost of labor and time is unavoidable in user

centered evaluation or expert evaluation. Therefore, this research builds a model-based evaluation method, focusing on improving its method performance while keeping the method cost low.

1.3 Two Phases of the Research with Machine Learning Methods

1.3.1 Reasons to Implement Machine Learning Methods

The most extravagant part of user centered evaluation method and expert centered evaluation method is that each time evaluator works, their work only applies to one specific user interface. Evaluators have to repeat their work even if "similar" usability issues have been encountered many times elsewhere. Expert evaluation is essentially an experience summary of those "similarities". This research can be treated as an attempt to implement computational models to help summarize and integrate evaluators' work. The focus is not to discover user interface design heuristics in a mathematical language, but to find the way about how to use data to find patterns of potential or unknown heuristics for user interface evaluation. Therefore, the needed implementation method should have the ability to discover implicit patterns from data, which is exactly machine learning method family's specialty. Machine learning methods refer to a group of stochastic methods widely used in different fields and have produced many successful applications. For example, using reinforcement learning and deep learning algorithm to play Atari video games, implementing advanced tree search and deep neural networks to play Go game, Image recognition and auto driven cars although are not mature enough for industrial application however have shown many gratifying improvements and are worth expecting. These promising results offer confidence to use machine learning methods in tackling problems of user

interface evaluation. More important, machine learning methods have two intrinsic similarities with user interface interaction. First, user interface interaction is a stochastic process. Different users have different ways to interact with the same user interface. Even the same individual cannot reproduce exactly the same interaction. Machine learning method is essentially a stochastic method to organize and analyze data. Second, the user interface evaluation process is a kind of pattern recognition problem. User interface evaluation is a process to label an instance of user interface in a desired way. Discovering patterns shared by the same label of instances is what machine learning method is good at. (Murphy, 2012)

1.3.2 Two Phases of the Research

This research contains two phases: Phase I static user interface evaluation, evaluating the satisfaction aspect of usability and Phase II dynamic user interface evaluation, evaluating the efficiency aspect of usability. Since satisfaction is defined as how pleasant users feel in viewing a user interface, static feature evaluation focuses on static features such as widget position, dimension and other design features of different types of user interface components. Dynamic interaction evaluation focuses on dynamic features such as operation smoothness and task completion time. In other words, Phase I and Phase II of the research correspond to two main functions of a user interface, namely information presentation and interaction.

Phase I modeling aims to build a satisfaction classifier to distinguish user interfaces with different satisfaction levels. Specially, given two user interfaces in the domain of this research, Phase I model can predict their satisfaction level and compare which one of them is more satisfying for the user group of training data. The whole process consists of two steps:

Step 1 (Phase I): Data collection

Data for Phase I modeling is used as the basis to discover user interface design patterns. It consists of a feature vector, which is used to describe features of an instance of user interface being studied, and a satisfaction score, which is a scalar quantity used to describe the satisfaction aspect of usability provided by participants' subjective ratings.

Step 2 (Phase I): Training process

The training process for Phase I modeling refers to implementing selected machine learning method and using collected data to tune the parameters of the corresponding classifier for stochastically best predicting and comparing satisfaction levels for a new instance of user interface.

Different from Phase I model's direct prediction on satisfaction level, Phase II modeling aims to first build an agent to interact with user interface mimicking human's behavior, and then using agent's interaction results as the basis to evaluate the efficiency aspect of usability with quantities such as task completion time. In other words, Phase II modeling simulates interaction on user interface first and then applying simulated results as the dynamic interaction evaluation's foundation. It could be asked why Phase II modeling does not implement the same classification method to directly evaluate interaction based on participants' subjective ratings. There are two reasons for it. First, as mentioned above about similarities between user interface interaction and machine learning method, user interface interaction can be regarded as a stochastic process. Performing tasks on a user interface, there does not exist correct or wrong interactions. There only exists interaction with high probability or low probability. Using one interaction result to evaluate efficiency of a user interface is not persuasive, and discovering the distribution of all the interactions for one user interface with one task is costly and time consuming. Therefore, building an interaction simulator becomes a good solution since it can generate a batch of

interaction results and the generated interaction results can be used to evaluate efficiency of usability. Second, an interaction simulator has generality in simulating interaction results that have not been shown before. The feature of generality can, to some degree, predict new possible usability issues. Phase II modeling also contains two steps:

Step 1 (Phase II): Data collection

Data for Phase II modeling is used as the basis to train agent mimicking human user behavior to interact with user interface. It consists of a task definition vector, a feature vector that is used to describe features of an instance of user interface being studied, and trace of interaction. Trace of interaction refers to a list of specific actions that the user would perform for a specific task.

Step 2 (Phase II): Training process

Training process for Phase II modeling refers to implementing selected machine learning method and using collected data to train agent's interaction with the user interface for the best mimicking of human user behavior.

1.3.3 Relationship between the Two Phases

Phase I and Phase II of the research, as mentioned above, evaluate the satisfaction and efficiency aspects of usability. It is difficult to combine the two phases into one for two reasons. First, user interface as a connection between the user and the machine has both the information presenting and instruction receiving functions. The two phases of the research correspond to the two functions of user interface, respectively. Relationship between the design requirement of a user interface to achieve the best information presenting function and to achieve the best instruction receiving function is still not clear. Second, there might exist correlations between the design requirement of user interface to achieve the best information presenting function and to

achieve the best instruction receiving function. For example, some designs of user interfaces are beneficial to both information presenting and instruction receiving. Therefore, it is difficult to use weighed factors to combine the two aspects of usability evaluations.

Phase I and Phase II of the research are not regarded as two evaluators for evaluating the two aspects of usability, efficiency and satisfaction, independently. They are actually two checkpoints to identify usability issues. For example, if the distance between two widgets of a user interface is too small or even overlapping, both Phase I and Phase II evaluators will provide low evaluation results for this issue. However, if the distance between two widgets of a user interface gradually increases, there must exist an interval of distance that high evaluation result shows up first in one of the two evaluators, since distance between two widgets for satisfaction evaluation is different from that for efficiency. In summary, the two phases of evaluators have their own emphasis on different aspects of usability but they are not necessarily fully independent of each other or cover all types of usability issues.

1.4 Practical and Scientific Values of the Research

From the practical perspective, this research aims to help evaluate and design user interfaces of computer based information systems. For example, today's application software development on computer based information systems always integrates many functions or task components into one user interface page. This function integration needs to be carefully evaluated to avoid usability issues, and the competitive field of software development requires an evaluation process with a short cycle. Phase I and Phase II of the research provide an efficient but not necessarily comprehensive usability evaluation tool to meet the demand of the field.

From the scientific perspective, this research aims to help researchers *make quantifiable predictions and evaluations* of user interfaces. Qualitative theories and models are important, but often insufficient *for rigorous understanding and quantitative analysis*. Therefore, this research work on computational model-based interface evaluation has *important theoretical value* in advancing the science of studying human behavior in complex human-machine-environment systems.

Chapter 2 Literature Review

Chapter Summary

This chapter first gives literature review about usability evaluation of user interfaces. Then, it briefly introduces machine learning methods that can be possibly used to build user interface evaluation models. Based on a comparison of these machine learning methods, it discusses the selection of machine learning methods for Phase I and Phase II user interface usability evaluation, respectively.

2.1 Literature Review on Usability Evaluation

2.1.1 Types of Usability Evaluation Methods

User interface usability evaluation methods can be categorized into three groups: user centered evaluation, expert centered evaluation and model based evaluation (Scholtz, 2004). The following part describes what these evaluation methods are in detail.

User centered evaluation usually refers to empirical usability testing, which mainly includes two types of evaluations: formative evaluation and summative evaluation. Formative evaluation is an informal evaluation method and is done by obtaining verbal data from users for early design of user interfaces. In the formative evaluation process, evaluation is commonly conducted through paper prototypes. Summative evaluation is a formal evaluation method and is done by documenting usability characteristics of user interface. In the summative evaluation

process, evaluation is commonly done through having representative users interacting with the user interface and discovering usability issues. User centered evaluation method involves participation of real users and thus is more possible to discover usability issues. However, the whole process of user centered evaluation is often expensive and time consuming. (Scholtz, 2004)

Expert based evaluation usually refers to usability inspection method as a group of evaluation methods that directly review user interface for discovering usability issues including heuristic evaluation, cognitive walkthroughs, pluralistic walkthroughs, formal usability inspections, consistency inspections, standards inspections, and feature inspections, guideline reviews. Expert based evaluation method is developed as a more efficient substitute for user centered evaluation method. In comparison with user centered evaluation methods, expert based evaluation methods are less expensive and less time consuming. However, it does not provide solutions to those discovered usability issues. Besides, the accuracy of expert-based evaluation methods is being doubted by some scholars (Scholtz, 2004).

Model-based evaluation methods refer to implement modeling techniques to simulate users' behavior interacting user interface and discover usability issues. GOMS model is one of the most famous model-based evaluation methods. GOMS model consists of goals, operators, methods and selection rules (Kieras, 1994). In comparison with user centered evaluation methods and expert evaluation methods, model-based evaluation method is less expensive and multiple rounds of usability testing can be done repeatedly. However, as indicated by Scholtz, task level cognitive task analysis is very time consuming (Scholtz, 2004).

Another classification way of user interface evaluation is complete empirical usability testing, which is effective but expensive, and the usability inspection method, which has lower cost by using expert review or direct analysis on user interface. Inspection method includes

expert evaluation method and model based evaluation method mentioned above. (Novick & Hollingsed, 2007)

2.1.2 Development of Usability Evaluation Methods

Early stage of Usability Evaluation

Researchers have started to pay attention to the quality of "user interface" several decades ago. At that time, it was under the name of ergonomics and focused mainly on the physical control panels (Shackel, Ergonomics for a Computer, 1959). In 1967, Michael Scriven developed formative and summative evaluation, which served as the basics for empirical usability testing. In 1971, the concept of ease of use was proposed but not well defined (Miller, 1971). After 1975, platform style guidelines appeared to aid the design process and some of them remain to be in continuous use nowadays. In 1979, the concept of usability was gradually formed. John Bennett published the first paper with usability in its title (Bennett, 1979). Some companies like IBM established usability labs to perform summative usability testing, and metrics for user performance gradually came into shape. In 1980, the think-aloud method was introduced into usability testing and became one of the most widely used methods (Ericsson & Simon, 1980). In 1983, GOMS model was developed and became one of the most widely known theoretical concepts in research on user interface interaction (Card, Newell, & Moran , 1983). GOMS model is a group of predictive models for user performance mainly used to evaluate usability and improve efficiency of user interfaces. GOMS model was then further adapted to different kinds applications and generated great effect to later model-based usability evaluation methods. In 1985, Jeff Kelley developed the OZ paradigm (now known as Wizard of Oz) method in his dissertation and came into widely use in the usability engineering as well as ergonomics and

psychology. From 1986 to 1988, models of iterative design, Motif style guide and SUMI QUIS were established in succession (Card & Moran , 1986).

Modern usability establishment

Shackel (1990) defined usability as efficiency, effectiveness and satisfaction. From the early 90s, the topic of usability attracted more widespread interests. Many researchers or scholars contributed to the field of usability testing. A large amount of usability evaluation methods is established and put into use. The rapidly developing field requires an effective method to evaluate user interface usability. A comparison between traditional empirical usability testing and "faster and cheaper" methods is widely discussed in the field (Jeffries & Desurvire, Usability testing vs. heuristic evaluation: was there a contest?, 1992; Desurvire, Kondziela, & Atwood, 1992; Jeffries, Miller, Wharton, & Uyeda, 1991; Desurvire H. W., 1994; Novick & Hollingsed, 2007). Traditional empirical usability testing evaluates user interface usability and discovers usability problems through observing actual users while they are interacting with a target user interface. Since recruiting real users to participate empirical usability testing is expensive and its whole process is time consuming, "faster and cheaper" evaluation methods showed up. "Faster and cheaper" methods usually refer to usability inspection methods as a set of cost effective ways of user interface evaluation (Nielsen, Usability Inspection Methods, 1994). Inspection methods, rather than discovering problems from observing real users, directly review user interfaces and discover usability issues based on experience or guidelines (Nielsen & Molich, Heuristic Evaluation of User Interfaces, 1990). Heuristic evaluation method was one the most widely used inspection methods. Heuristic evaluation is done by looking at a user interface and trying to discover usability issues based one's own opinion. Smith and Mosier (1986) proposed a large number of rules as user interface evaluation guidelines. In practice, usability heuristics

proposed by Nielsen and Molich as a substitute for empirical user testing are more commonly used. Nine user interface evaluation heuristics proposed in 1990 (Nielsen & Molich, Heuristic Evaluation of User Interfaces, 1990) are: (1) Simple and natural dialogue; (2) Speak the user's language; (3) Minimize the user memory load; (4) Be consistent; (5) Provide Feedback; (6) Provide clearly marked exits; (7) Provide shortcuts; (8) Good Error Message; (9) Prevent Errors.

The developers of these nine heuristics also mentioned the major advantages of heuristic evaluation, namely 1) low evaluation cost, 2) evaluation procedure is intuitive and easy to motivate participants to join, 3) evaluation does not require advance planning, and 4) evaluation can be performed in the early stage of development. They also designed four empirical tests to test the practical applicability of heuristic evaluation. Results indicate that heuristic evaluation is difficult to perform and researchers should not rely on individual's result. They pointed out that a disadvantage of heuristic evaluation is that it sometimes identifies usability problems without providing suggestions for improvement and the method is biased by the current mindset of the evaluators and normally does not generate breakthroughs in the evaluated design (Nielsen & Molich, Heuristic Evaluation of User Interfaces, 1990). Nielsen (1994) further refined and updated user interface evaluation heuristics to: (1) Visibility of system status; (2) Match between system and the real world; (3) User control and freedom; (4) Consistency and standards; (5) Error prevention; (6) Recognition rather than recall; (7) Flexibility and efficiency of use; (8) Aesthetic and minimalist design; (9) Help users recognize, diagnose, & recover from errors. The modified list of nine heuristics is based on factor analysis of 249 usability problems. Nielsen concluded that the modified heuristics succeeded in explaining discovered usability problems and its ability to discover new problems remains to be seen (Nielsen, Enhancing the explanatory power of usability heuristics, 1994). These nine heuristics remain to be used until now.

In addition to heuristic evaluation, there are also many other usability inspection methods. Some typical inspection methods are cognitive walkthrough, pluralistic usability walkthrough, formal usability inspection, feature inspection, consistency inspection and standards inspection (Nielsen, Usability Inspection Methods, 1994).

Cognitive walkthrough evaluates a user interface for its ease of exploratory learning through cognitive modeling (Wharton, Rieman, Lewis, & Polson, 1994). It simulates one's problem solving process and checks if this cognitive modeling can complete tasks or lead to correct actions (Nielsen, Usability Inspection Methods, 1994). Cognitive walkthrough consists of two phases: a preparatory phase and an analysis phase. The preparatory phase determines four key issues: interfaces, users, tasks, actions. In the analysis phase, evaluators go through the following four steps (Novick & Hollingsed, 2007):

1. set a goal to be completed within the system.
2. determine available actions.
3. select the action that leads closer to the goal.
4. perform the action and evaluate the feedback given by the system.

Cognitive walkthrough has some disadvantages: 1) It requires filling out evaluation forms repeatedly, 2) It can only find out limited range of problems, and 3) It has difficulty in defining availability of actions to users.

Pluralistic usability walkthrough modifies the traditional usability walkthrough and it involves representative users, product developers, members of the product team, and usability experts in the evaluation process (Bias, 1994). Pluralistic usability walkthrough is done by having a meeting of representative users, product developers, members of the product team, and usability experts to discuss user interface dialogue elements (Nielsen, Usability Inspection

Methods, 1994). This method does not have to be applied after user interface design is fully accomplished. On the other hand, the approach must be limited to representative rather than comprehensive user paths through the interface. The pluralistic walkthrough method appears to be in active use for assessing usability (Novick & Hollingsed, 2007).

Formal usability inspection combines heuristic evaluation and simplified form of cognitive walkthrough to implement six steps of procedures (Kahn & Prail, 1994; Nielsen, Usability Inspection Methods, 1994). It works faster, and it is more thorough and more technical than in the pluralistic walkthrough (Novick & Hollingsed, 2007). It appears that little research has been conducted on formal usability inspections (Novick & Hollingsed, 2007). Other usability inspection methods include feature inspection, consistency inspection and standards inspection, which only evaluate one aspect of usability issues and are less widely used in current user interface evaluation, and therefore they not illustrated in detail here.

Modern usability Development

After modern usability was established, the development in the usability field showed two main features.

First, research on usability gradually became mature and the system of usability testing became more rigorous. Research on the usability topic was divided in more detail gradually. Lewis examined experiment sample size for usability studies (Lewis, 1994). The concept of usability became a standard in ISO (International Organization for Standardization). Evaluators, as a key factor of usability testing, were also examined (Jacobsen & John, 1998). Some researchers summarized existing methods and made comparisons about these methods (Gray & Salzman, 1998).

Second, user interface as the subject of usability evaluation was dramatically influenced by software, Internet and portable devices. Three major changes after 2000 are listed below.

1. Software usability testing became the topic of research interest.
2. Usability testing data started to be collected remotely and analyzed stochastically.
3. Usability evaluation was influenced by massive use of portable devices.

These changes influenced usability evaluation methods through making new requirements to user interface design and further produced more usability evaluation methods.

As software development arose dramatically, software user interface became the main user interface to perform usability evaluation. The field made large efforts to adapt to this change. For example, in 1998, the Industry Usability Reporting Project was initiated by National Institute of Standards and Technology and many industry companies (NIST, 1998). The goal was to develop a usability reporting format with shared customer (software purchaser) data. More than 40 companies were involved in this project including some major manufacturers. This is a big effort in building standard for usability testing in the history. This project produced *Common Industry Format for Usability Test Reports* (CIF). In the past there were many different ways to report usability evaluation results. CIF became a very important effort to communicate between software developer and software purchaser. Almost at the same time of the Industry Usability Reporting Project, great development of Internet gradually changes the field of usability. After the year of 2000, Internet became popular in people's daily life. More people have access to Internet worldwide. It made remote user interface usability testing come true. Users started to participate from their normal work locations using their normal browser, and there was no real time observation helping to reduce performance difference between lab condition and real condition (Tullis, Fleischman, Mcnulty, Cianchette , & Bergel, 2002). After 2007, smartphone

and other portable devices went into market and dramatically challenged user interface design. As the different ways of interaction with computer based information system and various screen dimensions, these portable devices had very different requirements on user interface design as well as usability testing methods. After 2010, large amounts of applications on portable devices gave usability testing new challenges. Compared with lab based user interface usability evaluation, applications of portable devices went into and out of the market rapidly, which required usability testing to be conducted quickly and cheaply. Traditional usability testing methods still hold their scientific values in the field. However, it is also worthwhile to further develop practically useful usability testing methods to follow the rapid development pace.

In summary, the type, amount and interaction method of user interfaces that an individual use all had big changes in the past two decades. Correspondingly, many new user interface evaluation methods were built to adapt to these changes. These methods showed diversity on the whole and had the trend of becoming more quantitative and cost efficient.

Quantitative evaluation method for usability evaluation developed from many directions. Tom Tullis and Bill Albert (2008) rigorously studied user experience metrics and reviewed performance metrics, self-reported metrics, behavioral and physiological metrics, and so on, and provided instructive advice to evaluate through measuring user experience. Sauro and Lewis (2012) systematically studied and summarized how to implement statistic tools to evaluate user interface usability.

Cost efficient methods are mainly beneficial from widely used Internet and large data availability. Tom Tullis et al (2002) compared lab-conditioned usability testing and remote Web-based usability testing of Web sites and indicated that both the lab and remote evaluations capture similar information about the usability of a site. It built foundation of remote user

interface evaluation and show a possibility to make usability evaluation more cost efficient. Bill Albert, et al (2010) conducted large-scale online user experience studies, taking advantage of Internet and big data availability to improve cost efficiency of usability evaluation.

These new development in usability evaluation, cost efficiency and quantitative analysis met the recent years' flourishment of machine learning methods. The topic of "As Machine Learning and AI perform Magic, how can UX professionals help?" was discussed in User Experience Professionals Association (UXPA) conference in 2017. The discussion raised possible role change for UX professional in facing the rapidly developing AI field in order to 1) define overall user experience at strategic level; 2) understand the ecosystem that users interact with and the feedback loop for machine learning; 3) expand user scenarios for user research and design. Carol Smith (UXPA 2018) indicated that there were many remaining technical challenges to implement machine learning methods for user experience study. Therefore, this research will be an attempt to implement machine learning methods to evaluate the usability of user interface with the aim to build quantitative and cost efficient usability evaluation models with computerized procedures.

2.2 Literature Review on Machine Learning Methods

2.2.1 Overview of Machine Learning Methods

As mentioned above, rapidly developing machine learning methods provide a new possibility to build quantitative and cost efficient usability evaluation models with computerized procedures. However, machine learning methods refer to a group of probabilistic methods. To

select the proper machine learning method for usability evaluation, a literature review of candidate machine learning methods needs to be conducted.

Generally, machine learning methods can be divided into three types: 1) predictive or supervised machine learning methods; 2) descriptive or unsupervised machine learning methods; and 3) reinforcement learning methods.

Supervised machine learning methods aim to build mapping from inputs x to outputs y with given labeled pairs of input x and output y. These methods are always applied in classification such as email spam filtering, image classification, handwriting recognition and face detection.

Unsupervised learning aims to discover useful data patterns with given input data x. In comparison with supervised data, unsupervised data have to discover relationship from input data and find out data patterns underneath without labels of data. Unsupervised machine learning methods are usually applied in discovering latent factors, image inpainting, collaborative filtering and market basket analysis.

Reinforcement learning is different from traditional machine learning methods, supervised and unsupervised machine learning methods, in the sense that it does not learn from existing examples or data directly. Reinforcement learning aims to learn how to act or behave from environment feedback, reward or punishment signals, and is widely applied in tasks involving decision making. In comparison with supervised and unsupervised machine learning methods, reinforcement learning adjusts and forms mapping action strategies to accumulated rewards rather than discover or recognize potential data pattern. In other words, an agent of reinforcement learning keeps trying to interacts with the environment and learns to make decision of action that maximizes the total accumulative rewards (Sutton & Barto, 1998).

Phase I modeling of this dissertation research is a classification problem and the classification feature is clearly the usability evaluation score. Therefore, supervised machine learning method is the proper implementation method.

Phase II modeling of this dissertation research is to build an interaction simulator to mimic human users' behaviors. There exist two possibilities for machine learning method implementation:

1. Implementation of a supervised machine learning method through directly mapping user interface and task to the distribution of users' interaction traces. Mathematical representations of the user interface and the task together serve as a feature vector. Users' interaction traces serve as the data labels.
2. Implementation of a reinforcement learning method through regarding the user interface as the interaction environment and defining the tasks with proper set of reward functions.

Implementation of a supervised machine learning method on Phase II of the research theoretically works, but has many practical difficulties. First, task and user interface cannot form a mapping to a single interaction trace because interaction of human users is a stochastic process. It requires data set of very large size to tune parameters and train the model. Second, the label of each instance of the data is the distribution of interaction traces, which is multidimensional. To adapt multidimensional labels to many supervised machine learning methods will further increase the size of the required training data. Third, as the trace of interaction has randomness, it is hard to distinguish the classification results due to differences in user interface design or in individual users.

Implementation of the reinforcement learning method on Phase II of the research is therefore more suitable from the following three aspects. First, reinforcement learning is a

learning process from interaction. The goal of Phase II modeling is to learn human user's interaction. Second, user interface in reinforcement learning is treated as the interaction environment, which greatly simplifies the mathematical complexity of the model and also makes the model more flexible to adapt to new types of user interfaces. Third, reinforcement learning method would be less sensitive to the size of the training data with proper reward function set. For supervised machine learning methods, if not enough data are provided, failure of task completion might happen, but for the reinforcement learning method, only the human behavior mimic level is influenced and task completion will not be influenced. Based on these considerations, Phase II modeling of this research implements the reinforcement learning method.

2.2.2 Phase I Candidate Implementation Methods

Based on the discussion above, Phase I candidate implementation method should be supervised classification method. The following part of this section will go through the candidate implementation methods for Phase I modeling from the current widely used machine learning methods including K nearest neighbors algorithm (KNN), Naïve Bayes, neural networks (NN), support vector machine (SVM) and logistic regression (Murphy, 2012).

1. **K nearest neighbors Method**

   K nearest neighbors algorithm (KNN) is a non-parametric classification method. It classifies data points on the basis of K nearest data points' votes. Different settings of K value will lead to different classification results. Also, the vote weights can be adjusted to derive weighted adjusted K nearest neighbors algorithm (WAKNN). (Murphy, 2012)

2. **Naïve Bayes Method**

   Naïve Bayes method is a classifier based on Bayesian Theorem under the assumption that different features are strongly independent to each other. It classifies data

point with maximum likelihood estimation (MLE), which maximizes the likelihood function with given observations. (Murphy, 2012)

3. **Neural Network Method**

Neural Network method can also serve as the classifier. It consists of at least two layers and each layer consists of nodes. Each node connects with every node in adjacent layers using linear combination function and activation function. The first layer, also known as input layer, is used to input feature vectors; the last layer, also known as output layer, is used to output classification results. It classifies data point using neural network structures to approximate classification functions. (lan Goodfellow, 2016)

4. **Support Vector Machine Method**

Support vector machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. The algorithm produces an optimal hyperplane, which classifies new instances of data into categories given labeled training data. For example, in a two-dimensional space the separating hyperplane is a line. Data point falling on one side of the line belongs to one category. If the input data is not linear separable, kernelization tricks can be applied for classification. Kernelization tricks refer to mapping original data from input space to high dimensional feature space so that the linearly inseparable data can be linearly separated in higher dimensional space. One thing to notice is that increasing the dimensionalities will lead “the curse of dimensionality” since the required size of the training data increases exponentially with the dimensionality increasing. If there is not enough data provided, support vector machine model might be under the risk of overfitting (Murphy, 2012).

5. **Logistic Regression Method**

Logistic regression classifier is very similar to support vector machine classifier. The difference is that support vector machine implements hinge loss function and logistic regression implements logistic loss.

2.2.3 Phase I Modeling Method Selection

In the following discussion, four Phase I candidate implementation methods are compared and support vector machine method is selected as Phase I modeling implementation method.

In comparison with support vector machine method, K nearest neighbors algorithm has the advantage of having fast training speed, but has the disadvantage of not being able to handle many irrelevant features and having relatively low predictive accuracy. Naïve Bayes method, as mentioned above, needs to work under the assumption that different features are strongly independent to each other, which Phase I Modeling does not hold. Neural network and logistic regression methods are relatively more competitive to support vector machine method.

Neural network classifier has better predictive performance compared with support vector machine. A well-trained neural network classifier has higher predictive accuracy and faster predictive speed. However, it is not selected because of its two deficiencies. First, training a neural network classifier is very time consuming and relates to many complicated parameter tuning. Second, classification reason of neural network is harder to interpret. Support vector machine classification is also hard to interpret, but it classifies data points through maximizing the “gap” between different groups, and therefore separating hyperplane as classification boundary may provide more information to analyze classification reasons.

Logistic regression method as mentioned above is very similar to support vector machine method in mathematics and performance with a difference on loss function. Logistic loss

function of logistic regression diverges faster than hinge loss function of support vector machine and cannot reach zero under confident classification. In other words, logistic regression classifier is more sensitive to outlier data point. However, usability evaluation cannot avoid collecting data from participants with strong individual preference or bias. Support vector machine can better handle these outliers and reduce outlier's effect. Therefore, support vector machine is more suitable as the Phase I implementation method.

After comprehensive consideration of classification performance including accuracy and speed, robustness to usability evaluation data and modeling difficulty, support vector machine is selected as the implementation method for Phase I evaluation modeling. Detailed implementation for Phase I modeling will be introduced in Chapter 3.

2.2.4 Phase II Candidate Implementation Methods

Based on the discussion above, Phase II candidate implementation method should be in the reinforcement learning method family. The following part of this section will go through the candidate implementation methods for Phase II modeling, including Q learning method, state action reward state action (SARSA) method, deep Q network method, policy gradient method and actor critic method.

1. **Q learning Method**

   Q learning is one type of widely used reinforcement learning method. It includes a very important action-state mapping table called Q table. Q table is adjusted during the learning process to maximize expected accumulated total future reward for each action at each state. Columns of the Q table represent the actions. Rows of the Q table represent the states. The value of each cell will be the maximum of expected future reward for that given state and action. The action value function of Q learning method takes two inputs,

state and action and returns the expected future reward of that action at that state as output. Before the training process, the Q-table can be initialized with an arbitrary number. During the training process**,** the Q-table will be updated for better estimation of expected action value under state. The whole process can be summarized in four steps: 1) Initialize Q table; 2) Choose and perform action; 3) Receive feedback reward from environment; 4) Update Q table. After training process, action values on Q table can well estimate true action values therefore help agent make better decisions to maximize total expected accumulative rewards (Sutton & Barto, 1998).

2. State Action Reward State Action Method

State Action Reward State Action method (SARSA) is a similar reinforcement learning method to Q-learning. Difference between SARSA and Q-learning is that SARSA is an on-policy algorithm which means SARSA method updates the Q-value based on the action performed by the current policy rather than the greedy policy. Policy in reinforcement learning refers to a strategy to select a series of actions. (Sutton & Barto, 1998)

3. Deep Q Network Method

Deep Q network (DQN) can be regarded as a modification of Q learning method. The main difference is that DQN implements a Neural Network to estimate the Q-value function in substitute of the Q table in Q learning method. The input of the network is the state and the output of the network is the action value under the state. Besides, DQN is different with Q learning method from experience replay and separate target network which are used to reduce the effect of training sample's correlation and improve the stability of the training respectively. (Volodymyr Mnih, 2013)

4. **Policy Gradient Method**

Policy gradient method is a policy-based method. Rather than learning a value function, Policy gradient method uses the policy function directly updating the value policy. A policy function can be either deterministic or stochastic. A deterministic policy maps the state to action directly which is usually used in deterministic environments. A stochastic policy generates a probability distribution over actions. The stochastic policy is used when the environment is uncertain. (Sutton & Barto, 1998)

5. **Actor Critic**

Actor-critic method is a combination of value based method and policy based method. The policy and value estimation are done by actor and critic respectively. Critic criticizes the actions made by the actor. After each action selection by actor, critic evaluates whether the situation gets better or not. (Sutton & Barto, 1998)

2.2.5 Phase II Modeling Method Selection

Deep Q network (DQN) is selected as the implementation method for Phase II modeling. Five reinforcement learning methods mentioned in the previous section can be divided into two categories: value based method and policy based method. Generally speaking, policy based method is more appropriate to solve problems of high dimensional action space, since it is hard to build action tables. Therefore, it seems that policy based method is suitable for Phase II modeling implementation. However, after careful consideration it can be noticed that there exists a dilemma for policy based method. On one hand, in policy based method, policy is used as a whole to be updated and optimized to achieve a compelling goal, in the case of Phase II modeling which is task completion. At this time policy serves as a decision variable. However, on the other hand, task completion is not the only goal of interaction simulation of Phase II.

More important, interaction simulation of Phase II needs to mimic human user's behavior, otherwise task completion becomes meaningless. Therefore, policy also serves as the objective function. The solution of this optimization problem is the training data itself. It could be argued that it is possible to set task completion as objective function while setting human behavior mimics as a constraint. But unfortunately, quantitatively measuring the similarity between different traces of interactions is a more complicated topic beyond the scope of this research. If the similarity between different traces of interaction can be quantitatively measured, building interaction simulator then becomes trivial.

Since policy based methods are not suitable for Phase II modeling, the focus is turned to value based methods. Among the five candidate methods for Phase II modeling, only Q learning method and deep Q network method do not have any component of policy value estimation. And in comparison with Q learning method, deep Q network method is more powerful in solving more complex problems. Moreover, deep Q network method in comparison with Q learning method has better generality. For states that the Q-learning agent has not been encountered, it is difficult for Q learning to make good decisions. In other words, Q-learning agent does not have the ability to estimate values for unseen states. Neural network structure of deep Q network method approximates action value function, which enables the deep Q network agent to make a better decision for an unseen state.

After comprehensive comparison among the five candidate implementation methods for Phase II modeling, deep Q network is the most appropriate method to implement. Detailed implementation for Phase II modeling will be introduced in Chapter 5.

Chapter 3 Phase I Computational Modeling and Experiment

Chapter Summary

This chapter first introduces the objectives and challenges of Phase I evaluation model. Then it introduces the modeling scope, procedures and framework of Phase I evaluation model. Lastly, it described detailed implementation methods, including data collection and computational methods.

3.1 Objectives and Challenges of Phase I Modeling

3.1.1 Objectives of Phase I Modeling

Based on the discussions in Chapter 1 and Chapter 2, the purpose of Phase I evaluation model is to classify user interface based on the satisfaction aspect of usability with the selected support vector machine method.

User interface as an interaction media between human and machine involves two ways of communications. First, human user receives information from interface to process. Second, human user makes decision and gives instruction or acts on the user interface. In the first stage of communication, user interface serves its function of information presentation. Many usability issues could show up during this stage. User interface can be regarded as a static image. Through conducting research on features of the static image, the satisfaction aspect of usability can be evaluated.

Different from empirical usability testing, Phase I evaluation model does not include any actual dynamic interactions between the human and the interface. The scope of Phase I model is restricted to viewing and evaluating the **static features** of user interface. Phase II modeling will focus on evaluating a user interface based on dynamic interactions. It may be argued that when viewing a user interface, users would have already imagined how they would interact with a user interface. Therefore, static features of a user interface cannot be separated with evaluation of interactions. Truly, users might imagine how they plan to interact with a user interface before actually interacting with it. However, imagining to interact with a user interface is not the same as actually interacting with a user interface. For example, when widgets of user interface do not have clear boundary, it is easy to make mistakes even if it has been imagined successfully in the mind. Therefore, Phase I research is an attempt to capture usability issues from viewing interface as a static image.

3.1.2 Challenges of Phase I Modeling

During the implementation of support vector machine for Phase I modeling, three challenges are encountered.

First, user interfaces have too many feature inputs. User interface regarded as an image can be very complicated. To completely represent a user interface with a vector so that the image of user interface could be fully restored from this vector, all pixel values of the user interface need to be included. However, the size of the required training data increases exponentially with the size of the feature inputs. Implementing methods such as convolutional neural network to abstract useful information from pixel vector deviates from the main topic of this research. Therefore, it is necessary to manually select features as input training data.

Second, data points in the feature space of training data might be not linearly separable. There are two alternative solutions: soft margin support vector or kernel function mapping to high dimensional space. Soft margin support vector machine is a support vector machine allowing incorrect classification. The challenge is if collected sample is highly non-linear, it may reduce the prediction accuracy of soft margin support vector machine. The other alternative is to use kernelization trick to increase the dimensionality of sample space and classify data in high dimension space with corresponding hyperplane. It is safe to say that linearly non-separable data is dividable in a space with enough dimensions. However, this alternative has its own challenges, the choice of kernel function and the curse of dimensionality. There are many prevailing kernel function such as polynomial kernel, Gaussian kernel, radius basis function and sigmoid kernel. Different problem fits different type of kernels. If the choice of kernel can be solved by testing different kernels' results, curse of dimensionality is real barrier. Curse of dimensionality refers to the size of sample increases exponentially as the dimension increases which means implementation of support vector machine method increases required size of data to avoid overfitting. In dealing of this, it needs to either reduce the feature input or increase the size of collected data. Therefore, the dimension of the feature vector needs to be carefully designed based on the size of the training data.

The third challenge of Phase I modeling is the interpretation of the classification results. It is a common challenge for many other machine learning methods. Support vector machine classifies user interface through maximizing the "gap" between different groups. However, the classification results cannot directly benefit user interface design. As mentioned in method selection of Phase II modeling, it indicates that separating hyperplane as boundary of classification may serve as a breach to interpret classification results. Then, how to properly

interpret coefficients of hyperplane becomes a challenge.

3.2 Description of Phase I Modeling

Before describing Phase I model framework, it is necessary to briefly introduce the important components of Phase I modeling.

3.2.1 User Interface of Phase I Modeling

In the Introduction chapter, it has been mentioned that this research focuses on the visual user interface on computer based information system. And the discussion of Phase I modeling challenges mentions that to reduce the dimension of the feature vector, the number of features to be studied has to match the size of the training data. Therefore, it is necessary to make simplifications to the user interface in Phase I modeling.

Widgets on User Interface

A user interface consists of different functional widgets. Based on their functions, they can be divided into three types, as shown in Figure 3-1.

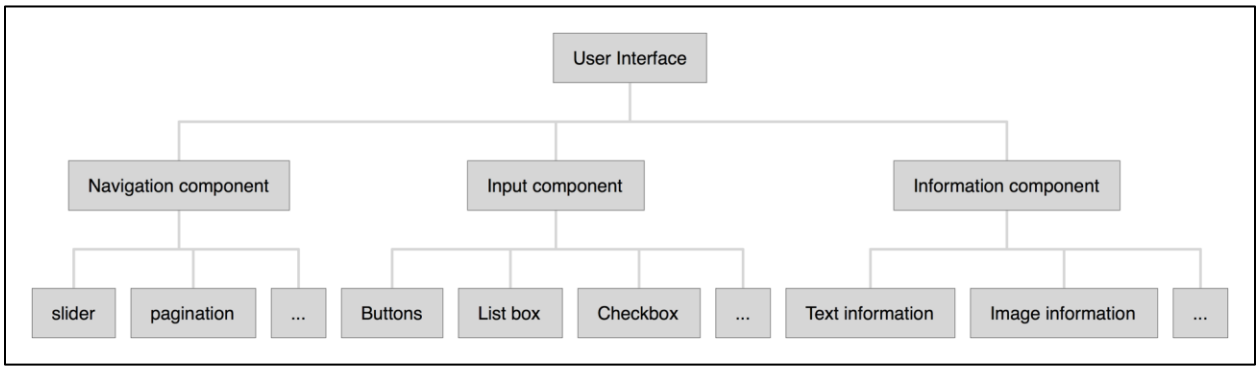


Figure 3-1: Different types of user interface widgets

Among all these different types of widgets, two widgets namely button and textbox are included. First, button and textbox are most commonly used widgets in all types of user interfaces. It is hard to see a user interface without these two widgets. Therefore, these two widgets are more representative than others. Second, button and textbox are two very intuitive

user interface widgets. It does not require training to know how to operate these two widgets. To reduce the number of variables, all widgets are in the shape of rectangle.

Framework of User Interface

The satisfaction aspect of usability is also influenced by the dimensions of user interfaces. Since Phase I of the research only evaluates from the static view of a user interface, related settings of user interface framework are dimensions and background color of user interface. The settings of user interface framework need to be as simple as possible to reduce the complexity of the Phase I modeling. The perceived dimensions of user interface framework are dependent on screen resolution, screen size and pixel size. User interface framework of Phase I modeling is 400×300 pixels on a 15 inches screen with 2880×1800 resolution. The background color is set as white to maximally reduce the effect of individual's preference.

Generation of User Interface

To collect satisfaction data for classifier training, sample user interfaces are needed for participants' subjective ratings. Generation of sample user interfaces consists of layouts of each widget. The central position of each widget is randomized within the range of the user interface framework. It may lead to widget's overlapping with the boundary of user interface, which can be used to study extreme conditions for the design of user interface static view. The dimensions of each widget are randomized within the range of 100 by 100 pixels. The reason to limit the dimensions of widget is to collect data of higher density in feature space in order to generate more statistics results, since the goal of Phase I modeling is to build satisfaction evaluation method, not to find out the optimal dimensions for user interface widgets. Generation of user interface does not avoid extreme conditions of user interface design such as overlapping but the generation process will guarantee at least part of each widget can be shown to the participants.

3.2.2 Other Components of Phase I modeling

Features

There might be many features or combinations of different features influencing the evaluation of satisfaction. When manually selecting features of a user interface, only the most important features can be included. Most important features should be able to decide the framework of a user interface and reflect a large proportion of static view of a user interface. Therefore, layout of the user interface including dimension and position of each widget is selected as features of interest for Phase I modeling. Features like color, which are strongly influenced by individual's preference, are not included. Each widget takes a four dimensional space in the feature vector, namely horizontal position, vertical position, horizontal dimension and vertical dimension. The total dimensionality of feature vector depends on the number of widgets on user interface.

Satisfaction Levels

Phase I model evaluates satisfaction of a user interface with a quantitative continuous scale ranging from 0 to 100, representing the satisfaction level from low to high. Since the datum line of different individuals might be different, absolute values of raw data need to be normalized in order to compare with each other. Normalization is performed with the satisfaction evaluation of the same five user interfaces. No user interface examples are provided to normalize satisfaction evaluation scores.

Implementation Methods

Support vector machine is selected as the implementation method of Phase I modeling. Feature vectors X and labels Y are used to train the classifier.

$$X = [x_1, x_2, \cdots, x_n] \quad \text{3-1}$$

$$Y = [y_1, y_2, \cdots, y_n] \quad \text{3-2}$$

Support vector machine classifies data points through a separating hyperplane. There may exist more than one hyperplane, which can separate training data. The "gap" between two groups of training data can be used as performance measurement for classification. This "gap" can be represented as the distance between hyperplane and closest point of data on both sides. Optimal hyperplane is the hyperplane that maximizes this distance. Support vector machine can be mathematically formulated as:

$$\frac{1}{m}\sum_{i=1}^{m} l(\omega \cdot x_i + b, y_i) + \left|\left|\omega\right|\right|^2 \quad \text{3-3}$$

$$Subject\ to\ min_i|\omega \cdot x_i| = 1$$

Feature vector X consists of layout of each widget and label Y refers to the satisfaction subjective rating.

3.3 Implementation of Phase I Modeling

3.3.1 Data Collection

38 participants from 18 to 26 years old University of Michigan students were recruited through emails in three batches to participate in the one-hour user interface evaluation study. Participants are from group of normal novice users. Recruited participants are required to have three years of experience of using any user interface on computer based information system. Participants should not have participated in the research before or have any usability evaluation experience. 3 subjects were initially recruited to test data collection user interface. 25 subjects' data were used as training data for the layout study and 5 subjects' data were used as verification data. Another 5 subjects' data were used to perform secondary study. Each subject was told to provide evaluation on 105 randomly generated prompted user interfaces with white background in gray scale of 400×300 pixels including button and textbox based on their satisfaction level of user interface. Each subject was required to drag an evaluation bar to indicate their satisfaction level to the user interface. Evaluation bar ranges from 0 to 100 referring to satisfaction level low to high. User interface of Phase I data collection is as shown in Figure 3-2.

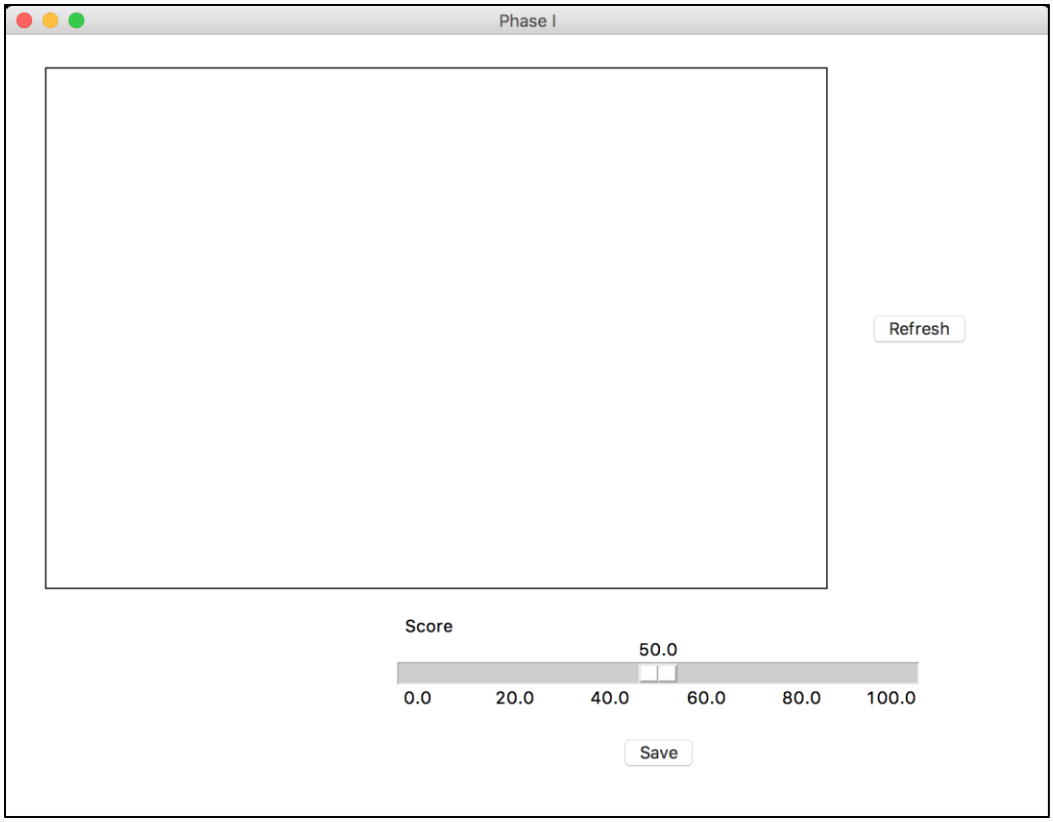


Figure 3-2: User interface for Phase I data collection

Collected data were divided into four cases. Each case contains different number of widgets as shown in Table 3-1.

Case Number	Number of widgets in user interface	Feature vector dimensionality	Number of user interfaces of the case
A	2	8	100
B	4	16	800
C	8	32	800
D	16	64	800

Table 3-1: User interface of Phase I model training data

For example, in case B the number of widgets is four. Each widget needs to be represented with horizontal position, vertical position, horizontal dimension and vertical dimension so that the total dimensionality of Case B is 16. 800 user interfaces with labels of

satisfaction are used as training data to tune parameters of classifier. An example of user interface for each case is shown in Figure 3-3.

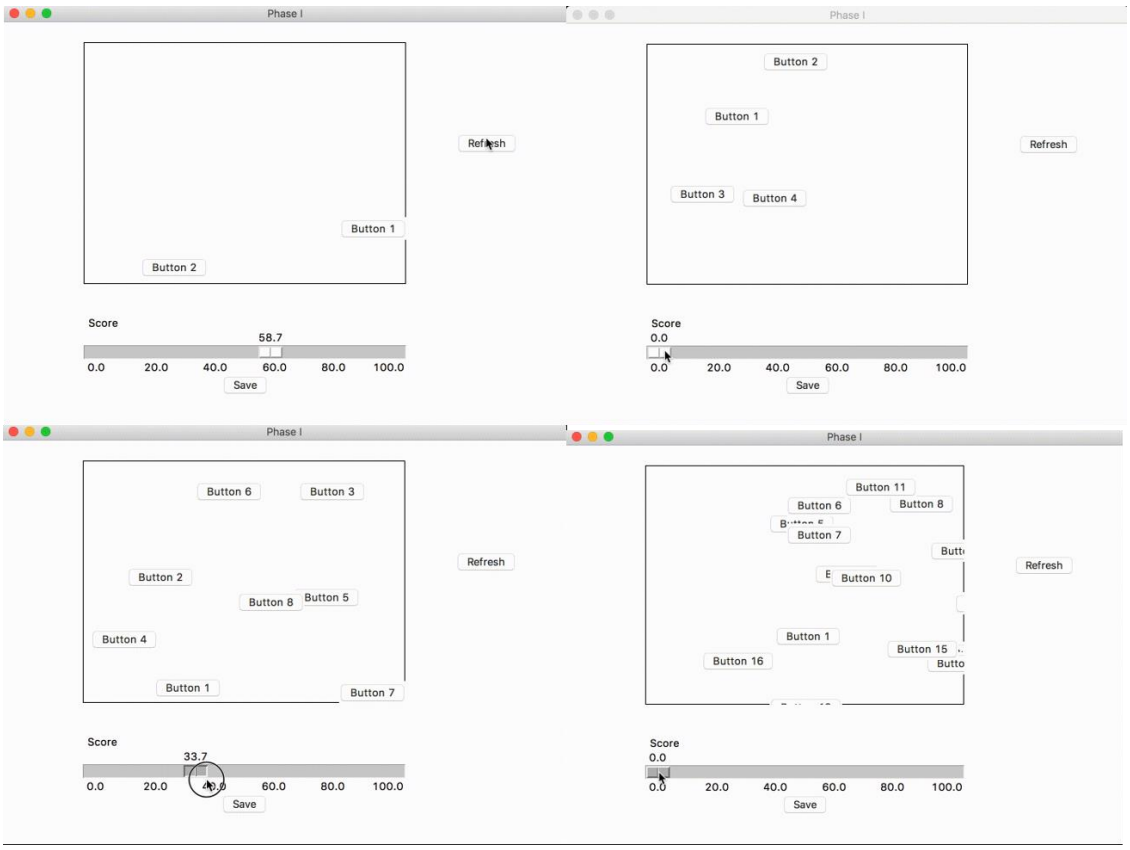


Figure 3-3: User interface example for four cases. Upper left: Case A; Upper right: Case B; Lower left: Case C; Lower right: Case D.

3.3.2 Normalization

At least five user interfaces are the same for all participants. These user interfaces' satisfaction ratings are used to perform data normalization and calibrate datum line of different participants. Participants are not told which user interfaces will be used for normalization. Normalized satisfaction evaluation results may exceed the upper limit of 100.

3.3.3 Model Training

Distribution Examination

Continuous scale of satisfaction evaluation provides freedom to select the position of classification. Before implementing support vector machine to classify, distribution of calibrated data are examined and possible classification positions are selected to insert a separating hyperplane. Separating hyperplane should avoid high density area since it cannot generate a well-defined boundary for different groups. Therefore, all the local minimum of satisfaction distribution are examined and classified with support vector machine training process. This is general principle to select classification boundary. In actual implementation of support vector machine, selection of classification boundary will be addressed on case basis.

Training Classifier with Support Vector Machine

Support vector machine can well classify linearly separable data points. For linearly inseparable data points, support vector machine needs to perform some adjustments based on the degree of linear inseparability.

Three methods of support vector machine family are tried in sequence to train classifiers. These three methods are:

1. Linear support vector machine
2. Soft margin support vector machine
3. Kernelized support vector machine

First, linear support vector machine method is always used first to classify linear separable data. If collected data is not linear separable, soft margin support vector machine is the next attempt. Soft margin support vector machine allows incorrect classification in training data. It has good classification performance with fuzzy boundaries. At last, if groups of data points not

only have fuzzy boundaries but also mixed together, kernel function is involved to increase the dimensionality of sample space. Kernelization tricks is powerful in making linearly inseparable data points linearly separable at cost of increasing required size of training data otherwise may lead to overfitting. Therefore, kernel function needs to be carefully selected and tested. There are many different types kernel functions. Phase I model makes the following kernel function attempts when implementing kernelized support vector machine method. These kernel functions are most widely used and best performance them will be selected to tune classifier parameters:

a) Linear kernel function:

$$K(\mathrm{v}_1, \mathrm{v}_2) =< v1, v2 > \quad 3\text{-}4$$

b) polynomial kernel function:

$$K(v_1, v_2) = (\gamma < v_1, v_2 > +c)^n \quad 3\text{-}5$$

c) Radius basis function:

$$K(v_1, v_2) = \mathrm{e}^{\gamma||\mathrm{v}_1 - \mathrm{v}_2||^2} \quad 3\text{-}6$$

d) Sigmoid function:

$$K(v_1, v_2) = tanh(\gamma < \mathrm{v}_1, \mathrm{v}_2 > +\mathrm{c}) \quad 3\text{-}7$$

e) Gaussian function

$$K(v_1, v_2) = e^{-\frac{||v_1 - v_2||_2^2}{2\sigma^2}} \quad 3\text{-}8$$

3.3.4 Verification

Verification study is conducted on four formal cases of Phase I evaluation model. Five participants' data are inserted to be tested in trained model. Classification accuracy rate is used to measure the performance of the classifier. Overfitting may be found during verification in view

of bad classification performance. In this case, support vector machine objective function may be adjusted through adding an extra term to penalize the number of parameters.

Chapter 4 Phase I Results and Discussions

Chapter Summary

This Chapter presents the training results of classifier and a discussion about Phase I modeling. It firstly presents classification results of four cases of Phase I evaluation model individually. Then a verification study of classification model is conducted. After verification, findings and discussions of Phase I modeling are introduced.

4.1 Classification Results

4.1.1 Overview of Classifiers' Training Results

Classification results are summarized in Table 4-1. After attempts to all the local minimum area of distribution of satisfaction ratings, two to three categories can be classified for all cases. Other possible classification positions are either inseparable with five kernelization methods mentioned above or of low predictive classification accuracy.

Case	Number of classified categories	Methods	Kernel function
A	3	Kernelized SVM	RBF
B	3	Kernelized SVM	RBF
C	2	Kernelized SVM	RBF
D	2	Kernelized SVM	Gaussian

Table 4-1:Summary of Phase I classification results.

From Table 4-1, it can be noticed that all the cases of classifiers implement kernelized support vector machine method, which means all the satisfaction data points are not linearly separable. After testing five candidate kernel functions, Case A, B and C implements RBF kernel function and Case D implements Gaussian kernel function to achieve best predictive classification accuracy. In the following, four cases of classifiers are discussed in detail.

4.1.2 Case A Classifier

Training data distribution of Phase I evaluation model of Case A is summarized in Figure 4-1. As mentioned in Chapter 3, all local minimum areas of distribution density are candidate places to insert separating hyperplanes.

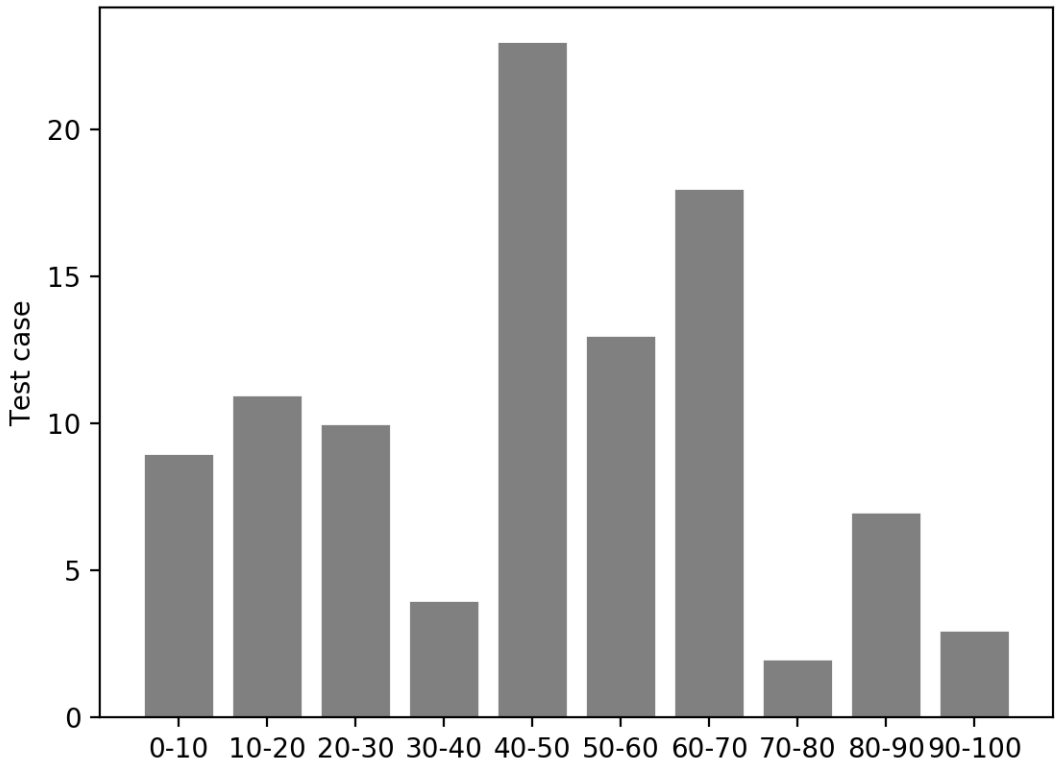


Figure 4-1: Case A data distribution. X-axis indicates usability evaluation ratings and Y-axis indicates the number of participants under some range of ratings

Therefore, support vector machine methods are implemented to classify the area around 30-40 and 70-80 intervals, since local minimum area of 50-60 has low predictive accuracy rate of verification. There are two possible options to implement support vector machine method.

1. Data points within 30-40 and 70-80 intervals are eliminated to avoid fuzzy boundaries between groups while tolerance value C of support vector machine is set to be less than 1 , which means the classifier allows mistakenly classified data.

2. It is also applicable to find two exact numbers within interval of 30-40 and 70-80 as classification position. The tolerance value C of support vector machine is set to be 1 ,which does not allow any mistakenly classified data around boundaries.

The advantage of the second option is the inclusion of two intervals' data and is more scientifically rigorous. However, since the measurement accuracy of subjective rating is unknown, it is hard to select one value within interval over others. Also, Inclusion of two intervals' data might not be able to improve the predictive classification performance but lead to fuzzy boundaries. Therefore, option 1 is easier to implement in comparison with options 2. Case A study implements option 1.

After the training of classifier, verification test is conducted. The verification data of Case A consists of 20 user interfaces. Among 20 test cases, four of them are mistakenly predicted and the prediction accuracy rate is 80%.

4.1.3 Case B Classifier

Training data distribution of Phase I evaluation model of Case B is summarized in Figure 4-2. Local minimum area is the interval of 10-20. Using the interval of 10-20 as the classification point may not be proper since in comparison with other intervals of data, its density value is high which might not provide a clear boundary. Also, if applying option 2 implementation method, exclusion of data within interval of 10-20 causes a large amount of data loss.

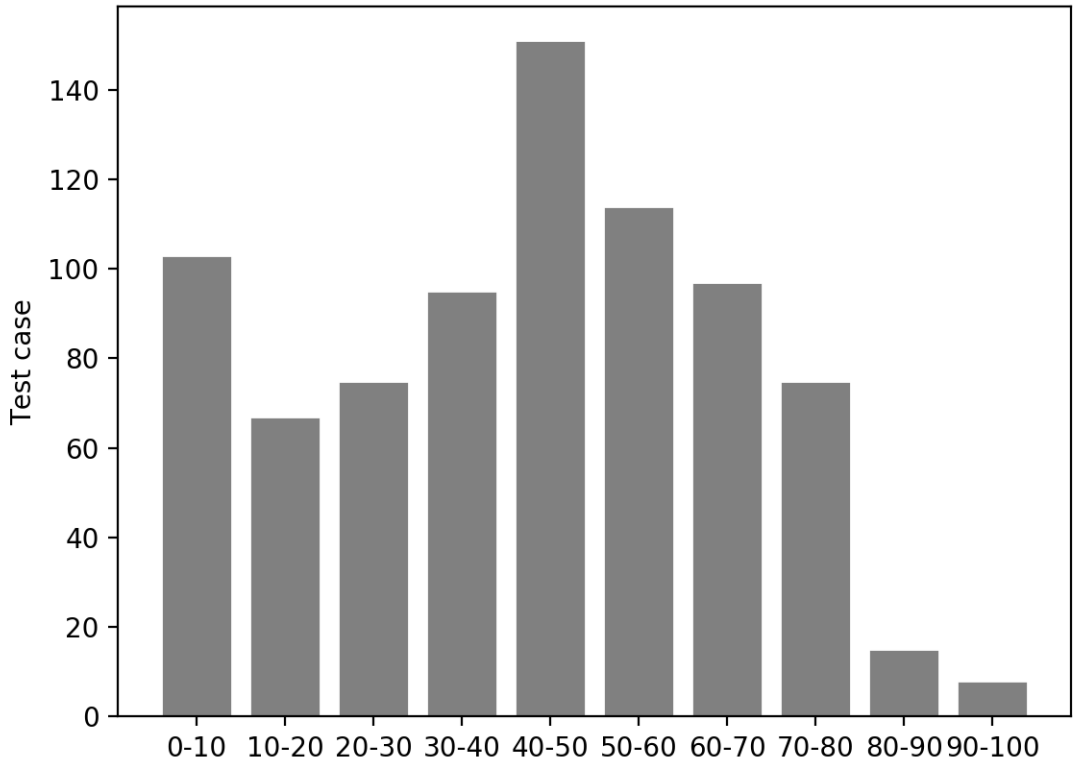


Figure 4-2: Case B data distribution

Considering the problems mentioned above, adjustment of classification position selection is made. In Case B, classification positions are selected within the big range of data sparsity namely long range of interval without data points in it. Similar to selecting local minimum areas, the selection of method also aims to provide clean boundaries between groups.

Based on the discussion above, separate hyperplane is placed at evaluation scores of 30 (28-33) and 80 (78-85). The first group ranges from evaluation score of 0 to 30. The second group ranges from evaluation score of 30 to 80. And the third group ranges from 80 to 100. Tolerance value C of support vector machine is set to be 1, which does not allow any mistaken classified data during the process of training.

After the training of the classifier, verification test is conducted. The verification data of Case B consists of 160 user interfaces. Among 160 test cases, thirteen of them are mistakenly predicted and the prediction accuracy rate is 91.9%.

4.1.4 Case C Classifier

Training data distribution of Phase I evaluation model of Case C is summarized in Figure 4-3. Local minimum areas are the intervals of 10-20, 50-60 and 80-90.

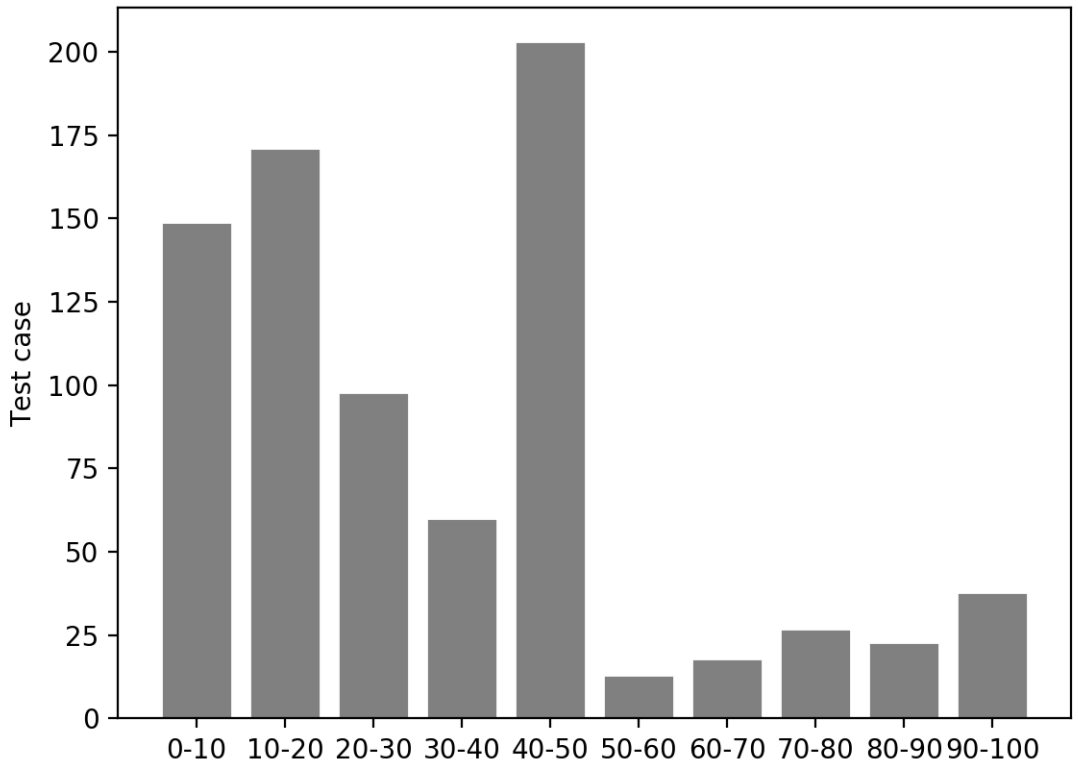


Figure 4-3: Case C data distribution

Though interval of 80-90 is a local minimum area, its density is about the same level as the both adjacent intervals. Therefore, separating hyperplane is placed at interval of 30-40 and interval of 50-60. Second option of support vector machine implementation is selected.

After the classifier is found, verification test is conducted. The size of verification data is 160. Among 160 test cases, 3 of them are mistakenly predicted. Therefore the prediction rate is 98.1%.

One thing to notice is that the distribution of Case C training data is right skewed. As mentioned previously, user interfaces of training data are generated through randomly layout widgets. With the number of widgets increases, the chance of generating user interface with high satisfaction evaluation becomes low, which also explains the asymmetric of Case C training data distribution. When the total number of widgets reaches 16, this asymmetric of training data distribution becomes more severe, Therefore, in Case D, extreme conditions including overlapping or edge cutting are manually eliminated.

4.1.5 Case D Classifier

Training data distribution of Phase I evaluation model of Case D is summarized in Figure 4-4. Local minimum areas are the intervals of 20-30, 40-50, 60-70 and 80-90.

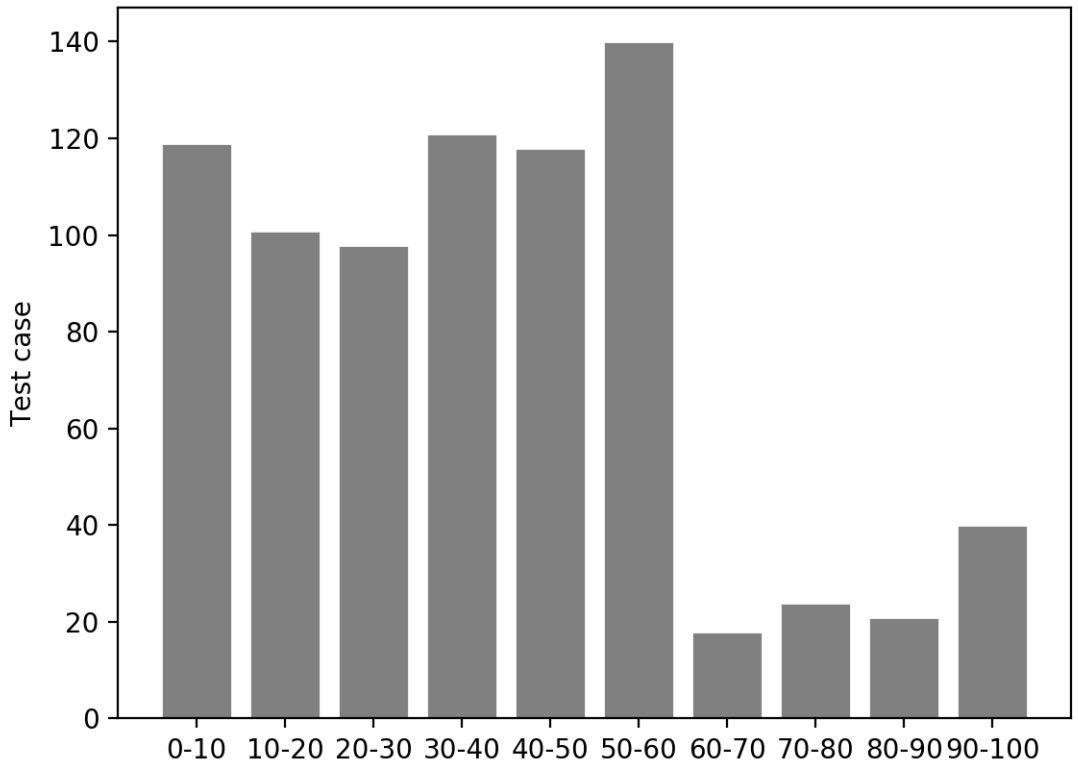


Figure 4-4: Case D data distribution

Though intervals of 20-30, 40-50 and 80-90 are local minimum areas, their densities are about the same level as the both adjacent intervals.

Separating hyperplane is placed at evaluation scores of 60-70. The first group ranges from interval of 0-60. The second group ranges from interval of 70 to 100.

After classifier is found, verification test is conducted. The size of verification data is 160. Among 160 test cases, three of them are mistakenly predicted. Therefore the prediction rate is 98.1%.

4.2 Findings and Discussions

In this section, findings and discussions of Phase I modeling are introduced from four aspects: data collection, training process, training result and implications of user interface design.

4.2.1 Training Data

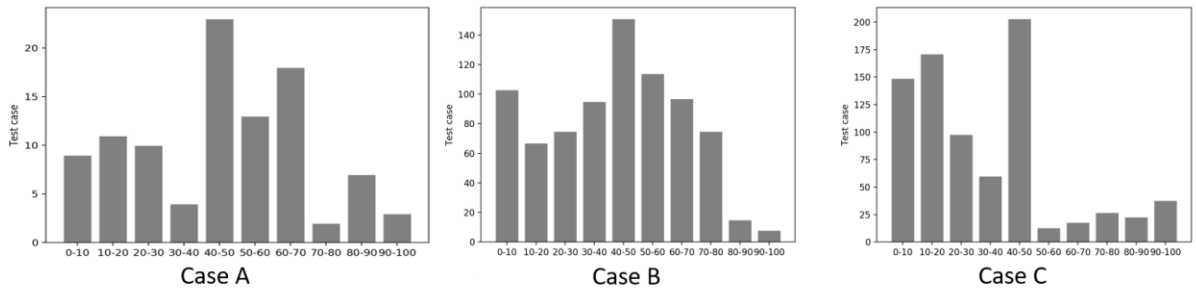


Figure 4-5: Case A, B and C satisfaction training data

In comparison with the distribution of collected data of Cases A, B and C, it is easy to notice that all these distributions are right skewed. Case D is excluded since extreme conditions of Case D are eliminated. Besides, with the increasing number of the widgets on the user interface, this right skewed characteristics becomes serious as shown in Figure 4-5. User interfaces of Phase I training data are all generated by computer through randomly laid out widgets on user interfaces, which means high satisfaction user interfaces are not uniformly distributed with the number of widgets on user interface. In other words, user interfaces with more widgets are harder to design. This fact raises two questions to researchers:

1. How to explain why high satisfaction user interface design with more number of widgets is harder to generate?
2. How to properly generate user interfaces for data collection to obtain symmetric data sets?

For the first question, a probabilistic model can be used to give estimation of difficulty level of user interface with different number of widgets. Considering design of user interface as a process of laying out widgets on a user interface, then for an empty user interface the probability to put first widget on an empty user interface within area of W_1 in order to maintain the satisfaction level $S \geq s$ is :

$$P_1 = P(w_1 \in W_1 | S \geq s) \quad 4\text{-}1$$

For the second widget, the probability to put second widget on an user interface with a widget within area of W_2 in order to maintain the satisfaction level $S \geq s$ is :

$$P_2 = P(w_2 \in W_2 | S \geq s, w_1 \in W_1) \quad 4\text{-}2$$

Therefore, the total probability to generate user interface with satisfaction level $S \geq s$ is:

$$\boldsymbol{P_S} = \prod_{i=1} P(w_i \in W_i | S \geq s, w_1 \in W_1, \ldots, w_{i-1} \in W_{i-1}) \quad 4\text{-}3$$

Since $P_i \geq P_{i+1}$, when the number of widgets increases, generation of high satisfaction user interface decreases even faster.

The biggest barrier for the second question is the lack of user interfaces with high satisfaction levels. In this research, it has been proved that randomly generation of user interface has low efficiency to produce user interface with high satisfaction level. It is necessary to have an algorithm to generate user interface while receiving feedback and assimilating experience from human participants. A possible future research could be combining support vector machine with reinforcement learning method. As mentioned in Literature Review, reinforcement learning method learns to act based on environment feedback. Generating user interface for participants is a process of laying out widgets on a user interface. Agent of reinforcement learning makes action to lay out widgets with different positions and dimensions. Support vector machine serves as

environment to provide feedback to agent. Relationship among participants, agent of reinforcement learning and support vector machine is as shown in Figure 4-6.

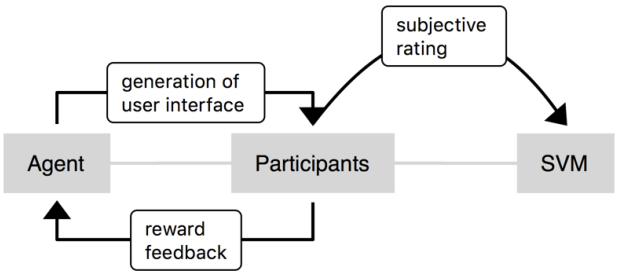


Figure 4-6: Future research for data collection

This will not only make the data collection process of support vector machine faster but also build a user interface designer with reinforcement learning method that can directly benefit to user interface design and serve as good tool to verify results of support vector machine.

4.2.2 Training Process

During the training process, it has been mentioned in implementation method that all local minimum areas need to be examined. However, during actual implementation of Case B, separating hyperplane is selected within the big range of data sparsity namely long range of interval without data points in it. This selection is essentially the same as the local minimum area method. They both aim to insert a separating hyperplane where it has lower density of data points so that the boundaries of different groups will be clean and easy to analyze.

4.2.3 Training Result

It is important to first discuss what have been done in Phase I research. In one sentence, Phase I modeling classifies the satisfaction aspect of usability with feature vector consisting of dimensions and positions. With simplification of user interface and limited number of user

interface features included, Phase I modeling is immature to be directly used for currently using real world user interface. The real value of Phase I model is to explore a way to use kernel functions to study the factors that may influence the satisfaction level.

Kernel function is used to solve non-linear classification problem through involving new dimensions. In other words, its essential meaning is to use kernel function to re-combine the features of interest so that the internal connections between different features will appear.

For example, in this research, Cases A, B and C of best performance with support vector machine RBF kernel function is not a coincidence. In the selection features to be included in feature vector, dimensions and positions are involved. RBF kernel function is a function whose values are only dependent on the distance to original point. With Euclidean distance, independent features of horizontal position x and vertical position y are combined into RBF function input $\sqrt{x^2 + y^2}$ which makes RBF function suitable to be used to implement distance related features of user interface. It also answers why RBF kernel function has the best classification performance.

This research only involves dimension and position features establishing possible connection to RBF function. In future research, more connections between user interface features and kernel functions can be established. It will be very beneficial from these connections. On one side, kernel function can be directly used to classify related features for user interface attributes of interest as is performed in this research. On the other side, this connection can serves as a basis to quantitatively study the design of user interface.

Another issue worth mentioning is the necessary number of categories used for usability subjective ratings. Although during the data collection, continuous scale of evaluation data is collected, it is very hard to find more than four gaps to insert separating hyperplane for more

detailed classification. Based on the classification results of this research, more than three categories of satisfaction subjective rating may be unnecessary. Some categories might be trivial or can be combined with adjacent categories.

4.2.4 Implications for User Interface Design

This section introduces some design implications based on the Phase I modeling results from general implications to detailed implications.

1. Avoid obvious usability issues

   From the classification results, it can be noticed that within classification group long range of satisfaction scores are not separable, which means as long as user interface does not have obvious usability issues normal users cannot sense the difference of small changes.

2. Widgets closer to upper left corner has higher satisfaction level

   As discussed above, Case A, B and C implements RBF kernel function, which is suitable to perform distance related feature analysis. Based on the classification results, smaller values of kernel function result higher satisfaction scores. Therefore, it can be inferred that widgets closer to upper left corner has higher satisfaction levels.

3. Features satisfaction sensitivities ranked from high to low are vertical dimensions, horizontal dimensions, horizontal positions and vertical positions.

   Based on the analysis of feature coefficients for all cases, sensitivities of feature satisfaction are ranked as above. It represents with each unit feature change how much it influences the satisfaction level. Coefficients of vertical dimensions are higher than horizontal dimensions in three of the four cases. Coefficients of horizontal dimensions

are higher than horizontal positions in all cases. Coefficient of horizontal positions and vertical positions are about the same.

Chapter 5 Phase II Computational Modeling and Experiment

Chapter Summary

This chapter introduces the objectives and challenges of modeling including research motivation and expectation. Then model description is introduced to provide a framework of Phase II evaluation. Lastly, it describes in detail the implementation methods including four components of interaction simulation, data collection and evaluation method.

5.1 Objectives and Challenges of Phase II Evaluation Model

5.1.1 Objectives of Phase II Modeling

Generally, the practically significant purpose or primary goal of user interface evaluation is to find out usability issues. In practice, most of the usability issues are found in observing interaction process between user and interface. Even for heuristic evaluation, most of its heuristics and design guidelines still need to learn and summarize observation results of user-interface interaction. Thus, it is believed that focusing on interaction process is a fundamental way to perform user interface evaluation. This does not seperate “interaction-based” user interface evaluation methods from others. But in the aspects of digging out new usability issues and adapting to different kinds of user interfaces, the “interaction-based” user interface evaluation is irreplaceable.

"Interaction-based" user interface evaluation is a subset of user-centered evaluation or empirical usability testing and therefore inherits the disadvantages that these methods share: expensive and time consuming. Cost of recruiting participants, performing experiment and analyzing feedbacks consumes a large amount of money and labor. It is needed to create a computational model that is capable of interacting with user interface and completing well-defined tasks on user interfaces to save the resources and shorten the evaluation period.

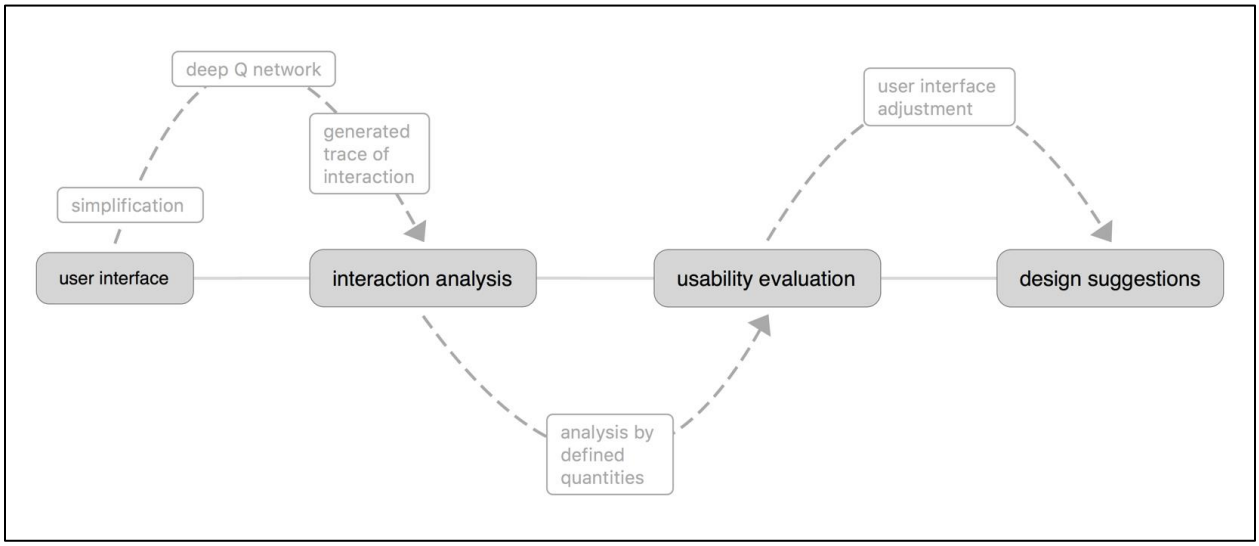


Figure 5-1: Objectives of Phase II user interface evaluation, interaction analysis, usability evaluation and design suggestions are organized in a hierarchy way. Interaction analysis serves as the foundation of usability evaluation. Design suggestions are based on the result of usability evaluation. Dashed lines represent the methodology applied to achieve each objective.

Specifically, the objective of Phase II user interface evaluation is to build a simplified and useful computational model that simulates user-interface interaction in a form suitable for analyzing human-computer interaction, performing usability evaluation and providing user interface design suggestions, as shown in Figure 5-1. As mentioned earlier, the most important part of user interface design is to understand the interaction process. Phase II evaluation method evaluates user interface through evaluating the interaction process. However, the interaction process is a complicated behavior involving many cognitive activities that have not been

thoroughly studied. Inclusion of too much details in modeling makes the generated result hard to analyze and assimilate. Phase II evaluation model, like other simulation models, generates prediction results in addition to producing evaluation results. The prediction result of Phase II evaluation model is the trace of interaction in time domain for a specific task. Trace refers to a list of actions that users perform for a specific task including all relative action reproduceable information for the task. Simulated interaction trace helps to observe and understand human-computer interaction process corresponding to the goal of "analyzing human-computer interaction". Performing usability evaluation requires the model being able to compare user interfaces with different levels of usability. Providing user interface design suggestions requires the model being able to actively suggest improved designs of user interfaces based on the original design. Suggesting good designs of user interface does not mean generating mathematically converged optimized design result but refer to improve usability of the original design of the user interface with adjustments based on the simulated trace of interaction.

5.1.2 Challenges of Phase II Modeling

There are three main challenges involved in achieving the objectives mentioned above. These challenges are not only barriers but also lead to the key contributions of this research. In the following, three challenges are introduced. How these challenges are solved will be addressed in the implementation of Phase II modeling.

The first challenge in Phase II evaluation model is to set up proper reward function considering both task completion and human factors. To apply reinforcement learning method, it is usually required to have a clear goal. For the majority of applications of reinforcement learning, there is a compelling reward or reward function based on this clearly defined goal to describe desired action or expected behavior. In user interface interaction, the goal is to complete

interaction task through cursor navigation and button click. However, setting reward function based on this straightforward goal will lead to a "perfect" interaction since discounting factor in reinforcement learning method leads navigation path to a shortest path. Different from many other reinforcement learning applications, Phase II evaluation model does not aim at improving the performance of machines in interaction tasks but acting imperfectly as real humans. In the human-interface interaction, humans do not always perform the best strategy or policy to complete a task. Cursors might go along with an irrational strategy a trivial loop or pass the target button without pressing. How to properly involving or modeling those human factors becomes the first barrier of the Phase II research.

The second challenge in Phase II evaluation model is how to describe the influence of different widgets on human interface interaction. User interface consists of different widgets. Those widgets work together and produce effects on human's behavior. Size, position and other physical characteristics might cause different influences. What makes it more complicated is that the influence of these widgets might change over time. For example, when target widget is not decided, all widgets impose the same amount of influence on human's interaction. Gradually, when users figure out the target widget, the influence of target widget may increase and that of non-target widget may decrease. Thus, second challenge requires modeling proper distribution of rewards describe single widget's influence on user interaction behavior.

The third challenge in Phase II evaluation model is how to perform evaluation with simulated interactions. After generating simulation interaction trace of user interface, there is no existing method to extract information from interaction as a base for user interface adjustment for usability improvement. The simulated trace of interaction now serves as raw input for user

interface evaluation. The problem is how to properly manage these traces of interaction to evaluate the efficiency aspect of usability.

5.2 Model Description

5.2.1 Overview of Phase II Modeling

User interface interaction involves a two-way communication between human and user interface. Human users take information from user interface as an input, cognitively process the information and generate actions on user interface as output. There are many researches working on bridging input information and output actions through modeling related cognitive activities and simulating and predicting human interface interaction such as MHP and ACTR modeling. A question is then raised that whether it is possible to directly bridge the gap between input information and output actions minimizing cognitive modeling in simulating user interface interaction as shown in Figure 5-2.

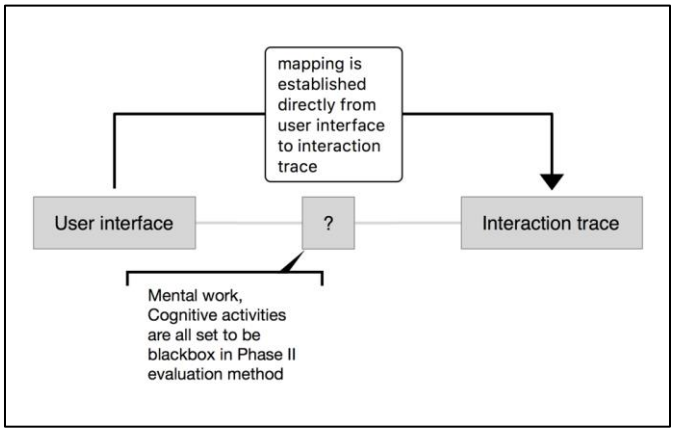


Figure 5-2: Bridge the gap between user interface and interaction directly

Cognitive modeling has special advantage in explaining what happens in human's mental world but does not take full advantage of observed interaction results. In Phase II evaluation model, it minimizes cognitive modeling and only cares observed interaction results. It believes that different levels of cognitive activities will show difference in interaction results. For example, if

hesitation occurs during interaction, cursor shows different path pattern with that of normal use. Therefore Phase II model simulates interaction from user interface and task directly without including cognitive modeling.

5.2.2 Assumptions of Phase II Modeling

Phase II research selectively makes some proper assumptions and reasonable simplifications to enable model to generate useful results. In the following, it will introduce assumptions the Phase II modeling.

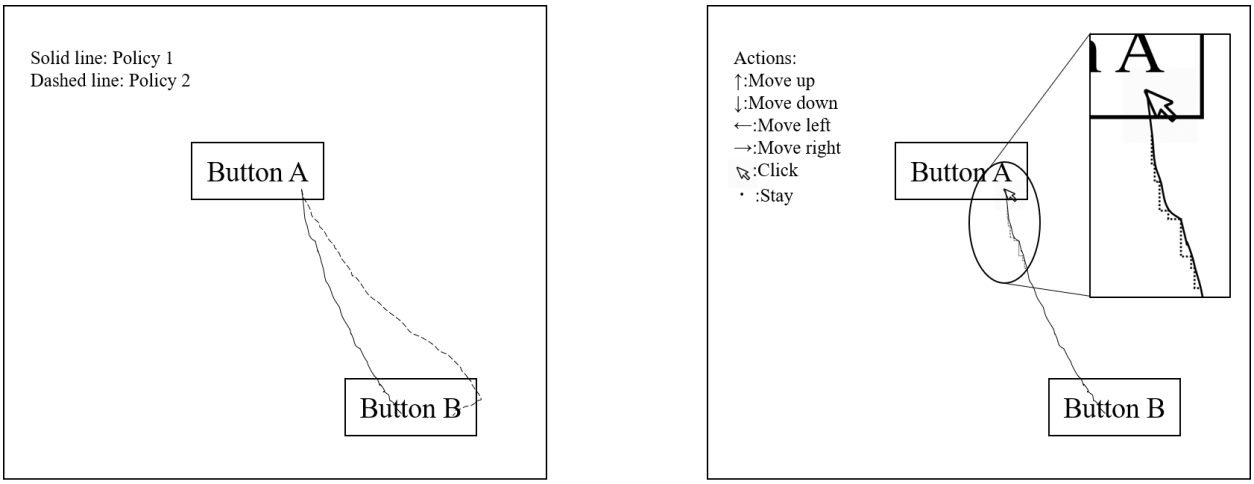
The first assumption of Phase II evaluation modeling is that user interface interaction is assumed to be a Markov Chain Process characterized with memorylessness. Future user interface interaction is only determined by current state of interaction. Previous interaction path and actions will have no effect on future interaction path and actions. Mathematically shown as equation 5-1,

$$Pr(\mathrm{X_{n+1}} = \mathrm{x} \mid \mathrm{X_1} = \mathrm{x_1}, \mathrm{X_2} = \mathrm{x_2}, \cdots, \mathrm{X_n} = \mathrm{x_n}) = Pr(X_{n+1} = x \mid X_n = x_n) \qquad (5\text{-}1)$$

This assumption practically fits the feature of user interface interaction. User interface interaction is a goal orient task. Users' past interaction will not influence the way how they complete task in the future. In other words, users do not try to avoid or repeat their past actions and do not care about these past actions at all. They interact with user interface and achieve target goal as if cursor is initially in the current position. Therefore, in Phase II evaluation model next action is only determined by displayed user interface and current cursor position.

The second assumption is in Phase II evaluation model user interface interaction is assumed to a discrete time task. When modeling trace of interaction, there are two possibilities, in continuous time domain and in discrete time domain. Continuous time modeling has advantage in researching on interaction policy. Discrete time modeling has advantage in

researching on action value or state value of interaction. Policy refers to strategy to complete a task. Action refers to each step of completing a task. For example, for a task of navigating from Button A to Button B on a user interface, Figure 5-3



a b

Figure 5-3: Left figure a) refers to continuous time interaction model and right figure b) refers to discrete time interaction model.

In continuous time model, each policy is indivisible is the smallest unit to research on. Navigation traces, solid line and dashed line, from Button A to Button B shown on Figure 5-2-a are two specific policies to complete task. Human operators take navigation from Button A to Button B as one output action. In discrete model, interaction is decomposed to five navigation actions: move up, move down, move left, move right, stay and a decision making action: click. If the step width of navigation action is small enough, the navigation path of discrete time model has negligible difference with that of continuous time model. And discrete time model has advantages over continuous model in modeling user interface interaction. Continuous model focuses on interaction policy and try to research on human interaction pattern on policy level. Under the condition of lack of enough experimental data, it cannot obtain reliable results. Also, staying on policy level to research on interaction pattern ignores many details of interaction. If an

interaction trace is regarded as a good result of interaction then the whole trace of interaction is regarded to be good but what part of interaction trace really works and contributes cannot be analyzed.

The third assumption of Phase II evaluation model is that interaction between human and interface is a series of goal oriented careful actions, which means it does not include aimless operations on user interface. In real world use of user interface, it happens that human users generate irrational or non-goal oriented actions. An example of non-goal oriented action was found in early stage of data collection. At the beginning of data collection, a user interface including cursor is shown to participants and asks participants to complete tasks. After completion of the task, participants repeat this process for multiple rounds. The problem happens each time when user interface was just shown to participants. Some participant has a habit of shaking cursor in order to find cursor position. This shaking action is irrelevant to user interface itself and somehow influences learning process of agent's interaction. In the later experiment of user interface interaction, cursor position is always shown to participants to avoid unnecessary actions. Therefore, in the evaluation model of Phase II, we try to only model interaction related to tasks. More details will be mentioned in experiment design.

Two user interface interaction issues excluded from Phase II evaluation model are also worth mentioning.

1. Error: Phase II evaluation model does not model errors, including errors due to decision making and errors due to misconduct. Errors due to decision making always involves complicated cognitive activities. Errors due to misconduct might be related to user interface design. However, this design issue can be reflected through observing interaction result of successful completion of task.

2. Linguistics and semiotics: Labels of widget plays an important role in user interface design. In the user interface design, language and symbol are most commonly used to label a widget. Human users need to understand these labels to be able to properly operate on a user interface. However, understanding label and making decision based on label content requires a series of cognitive processing. Linguistics and semiotics are too big topics to be discussed. Research about what are proper labels for a user interface widget to improve usability is beyond the scope of this research. Therefore, in the scope of the Phase II evaluation model widget is only labeled with one capitalized English letter avoiding influence from linguistic and semiotics aspects.

5.2.3 Structure of Phase II Modeling

Figure 5-4 shows the model structure of Phase II evaluation method. From the user interface to user interface evaluation, it goes through three main steps of interaction simulation, quantity analysis, user interface evaluation and manipulation.

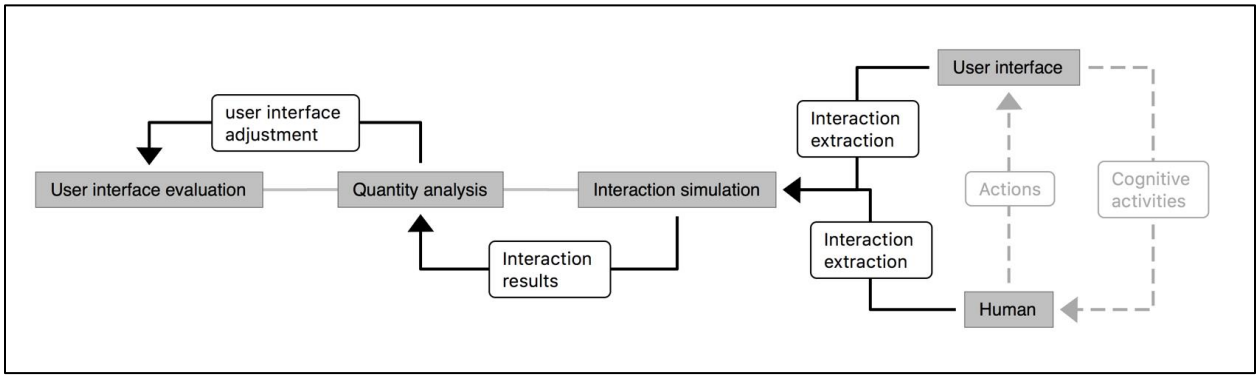


Figure 5-4: Phase II evaluation model structure.

In interaction simulation, an agent learns to use interface through interaction and feedback from environment. The motivation of agent is to maximize total accumulative rewards. Rewards describe the expected behavior of agent. In Phase II evaluation model, expected

behaviors are first completion of task and second in a similar way to human users. The essential part of interaction is to properly set reward value to guide agent's behavior.

In the quantity analysis, simulated user interface interaction results are being analyzed to compare usability of different design of user interfaces. Since there is no single metric to measure all aspects of usability, Phase II evaluation model implement different quantities and collect information from simulated user interface evaluation results to evaluate one aspect of a user interface usability.

In the user interface manipulation, widgets of user interface are being manipulated at the aim of improve usability of interface based on quantities mentioned above. Since those quantities can only reflect one aspect of user interface usability and many factors of user interface design are highly correlated, unrestricted changes made to user interface may lead to extreme conditions. Therefore in user interface manipulation, restricted manipulation to user interface aims to receive usability improved interface. Unrestricted manipulation can help to find special design of user interface or innovate the future of user interface design.

5.2.4 Components of Phase II Modeling

In this section, different components of Phase II modeling are introduced. User interface and type widgets used in Phase II modeling are the same as Phase I modeling. Therefore, it is not repeated here.

Different from Phase I evaluation method, Phase II Model emphasizes the importance of interaction during the user interface evaluation. The whole process does not directly implement human evaluation results as the basis but mimics human users' interaction results as the evaluation basis, which makes it possible to jump out of the box evaluating a user interface. The

core idea of reinforcement learning is trial-error learning. This process is similar to the learning process of humans. As indicated by Thorndike:

> *"Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond" (Thorndike, 1911, p.244)*

The learning process of reinforcement learning does not provide instructions directly. However, it adjusts or corrects agent's behavior through a reward function and a value function.

Starting with an agent interacting with a user interface and trying to complete task of clicking button A as shown in Figure 5-5. The Agent can choose one of six actions: Move up, Move down, Move right, Move Left, Stay and Click each round of action. Agent needs to firstly navigate to button A and secondly click button A to complete the task. If modeled with Q learning method, reward values are shown in the Table 5-1. State of the agent refers to cursor position. Then it has a size of 400 by 300 pixels has $256^{400\times300}$states. Under each state, agent can perform 6 actions. The framework of Q table is shown in Table 5-2. Each cell in Q table refers to action value of action a, $Q^{\pi}(s,a)$, under policy π at state s. Action value of action a represents for value to choose action a under specific state.

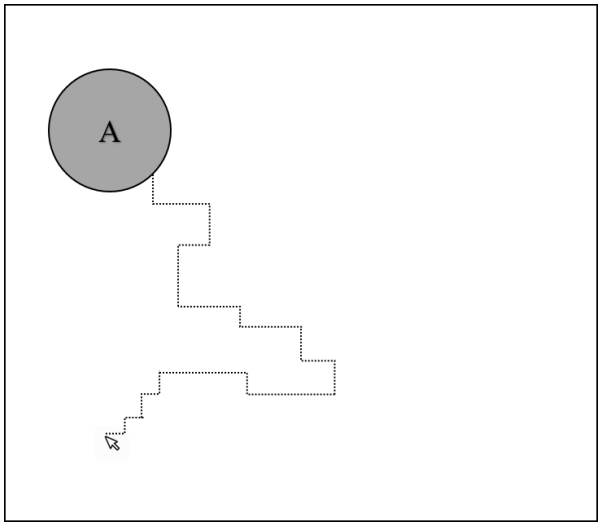


Figure 5-5: Agent learns to interact with user interface. Without interaction data, interaction is like lattice random walk.

Move Up	Move Down	Move Right	Move Left	Stay	Click
0	0	0	0	0	+1: within gray circle -1: outside of gray circle

Table 5-1: Reward feedback from environment for a simple user interface interaction.

States	Move Up	Move Down	Move Right	Move Left	Stay	Click
1						
2						
…						

Table 5-2: Q table for interaction with user interface with one button.

$Q^{\pi}(s,a)$ can be represented as the expected total rewards of future rewards.

$$Q^{\pi}(s,a) = \boldsymbol{E}[r_{t+1} + \lambda r_{t+2} + \lambda^2 r_{t+3} + \cdots | s,a] \quad (5\text{-}2)$$

λ is discounting factor to indicate action value discount in future. λ ranges from 0 to 1. The larger λ is, the more probably agent traverses to high reward position directly. The smaller λ is,

the more probably agent traverses as many as non-zero reward position. Using Bellman function, action value function can be expressed as

$$Q^{\pi}(s,a) = \boldsymbol{E}_{\boldsymbol{s}'}\left[r + \lambda\, Q^{\pi}(s',a')|s,a\right] \tag{5-3}$$

During the process of interacting with user interface, agent updates Q table to find out true action value under each state.

$$Q(s_t,a_t) \longleftarrow Q(s_t,a_t) + \alpha\big(r_{t+1} + \lambda \max_a Q(s_{t+1},a) - Q(s_t,a_t)\big) \tag{5-4}$$

$\alpha \in (0,1)$ refers to stepwise to update Q table since during learning process expressing $Q^{\pi}(s',a')$ with $max_a Q(s_{t+1},a)$ is an approximation. Therefore, each time update Q table, action value only moves towards true value by step width α. Each step of selecting actions, $\epsilon - greedy$ is used for policy π. Under this policy, agent has probability of $1-\epsilon$ to select actions based on current Q table and probability of ϵ to select action randomly. This policy consider both of the exploration and exploitation. Exploitation keeps learning from interaction to obtain more accurate Q table value and exploration helps to update Q table and discover new policy.

Modeling mentioned above is prototype at the beginning Phase II evaluation modeling. It generated some results and also reminds some places to improve. There are two important issues worth mentioning. Firstly, interaction model mentioned above has way too many states of interaction. It includes user interface into interaction states. On a user interface of gray scale with $n \times m$ pixels, each pixel values from 0 to 255 in total of 256 possibilities. Then the total number of states reaches $256^{n\times m}$. It may need to involve convolutional neural network to help pattern recognition which is very consuming. Its advantage is if successful it can provide a general interaction agent with all user types interfaces but also requires large computational capacity and sufficient data to support it. Therefore, rather than including user interface design into states of interaction, only cursor position is included in states of interaction and user interface is regarded

as interaction environment. Apparently, this model setting has disadvantage that every time user interface changes model needs to repeat learning process. However, it greatly reduces the computational workload. Secondly, it can be noticed that agent is a strictly rational and goal oriented operator. It hunts to gain more accumulative rewards. Except for providing positive reward feedback to agent for completing interaction task, it also needs to provide positive reward feedback to agent for actions that mimic human's behavior. Otherwise, it could happen that providing proper discounting factor λ agent should theoretically be able to complete interaction task in a most efficient but robotic way which very rare human users navigate in. Phase II evaluation model is built based on prototype above. It has two parts: interaction simulation method and evaluation method. Interaction simulation serves as basis of evaluation and consist of four main components: action, reward function, value function and model of environment. We will firstly introduce four components of interaction simulation

Model of Environment

Phase II evaluation model takes pixel value of user interface as raw input in gray scale. Each pixel value ranges from 0 to 255. Model of environment is user interface with its widgets and does not include content information labeled on these widgets as shown in Figure 5-6.

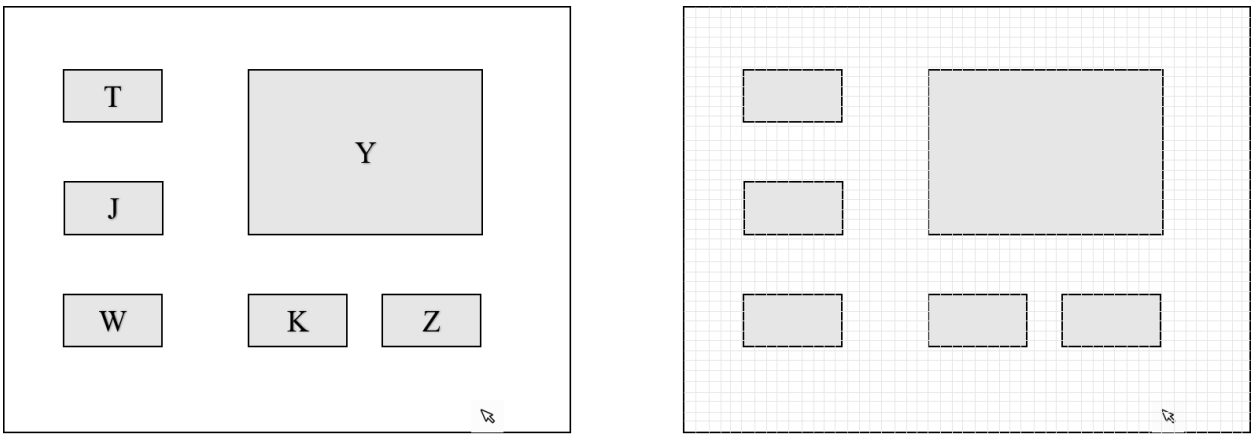


Figure 5-6: User interface as the model of environment is a collection of user interface image pixels excluding content of widget labels. Model of environment can fully represent what users see on a user interface.

Figure 5-6 left-side represents for original user interface and Figure 5-6 right represents for model of environment for agent to interact on. User interface has white background and each widget has clear border to indicate clickable area (to avoid the same color of background and widget and clickable area is not clear). Model of environment can be mathematically represented by a matrix of U. The dimension of U is determined by the size of user interface. Each element of U is the pixel value of user interface image. Usually matrix norms are used to measure the distance of matrices such as $p - norms$ shown as equation 5-5.

$$||\cdot||_p = \sqrt[p]{\sum_{i=1}^{n}\sum_{j=1}^{m}\left(U_{1_{ij}} - U_{2_{ij}}\right)^p} \tag{5-5}$$

Matrix norms are commonly used to reflect similarity between two interfaces. However, norms provide absolute values of distance between matrix. In Phase II evaluation model, we are looking for a percentage value of similarity. Therefore, we use a cosine value of matrix to describe the similarity between user interfaces as shown in equation 5-6.

$$I = \frac{\left(\sum_{i,j} U_{1_{ij}} \cdot U_{2_{ij}}\right)^2}{\left(\sum_{i,j} U_{1_{ij}}^2\right) \times \left(\sum_{i,j} U_{2_{ij}}^2\right)} \quad (5\text{-}6)$$

If user interface $U_1$and U_2 are exactly the same, then I is equal to 1. The more different $U_1$and U_2 are, the closer I is to zero.

Actions

Agent has six possible actions to interact with user interface, Move up, Move down, Move left, Move right, Stay and Click. Agent interact with model of environment is under discrete time domain. Each step or unit of time, agent can choose one of the six actions. Each action made might cause state change and receive rewards. Expected actions made lead to positive rewards and unexpected actions made lead to negative rewards. Detailed information of actions is summarized in Table 5-3. Table 5-3 summarizes possible changes of state and possible rewards by each action.

Actions	State	Reward
Move up	1. Horizontal position keeps the same. 2. Vertical position -1 pixel to new state if current position of cursor is not adjacent to the upper border of user interface; vertical position keeps the same if current position of cursor is adjacent to the upper border of user interface. 3. Interaction continues	Positive rewards proportional to reward density at next state from collected human user interface interaction data.
Move down	1. Horizontal position keeps the same. 2. Vertical position +1 pixel to new state if current position of cursor is not adjacent to the lower border of user interface; vertical position keeps the same if current position of cursor is adjacent to the lower border of user interface. 3. Interaction continues	Positive rewards proportional to reward density at next state from collected human user interface interaction data.
Move Left	1. Vertical position keeps the same. 2. Horizontal position -1 pixel to new state if current position of cursor is not adjacent to the left border of user interface; vertical position keeps the same if current position of cursor is adjacent to the left border of user interface. 3. Interaction continues	Positive rewards proportional to reward density at next state from collected human user interface interaction data.
Move Right	1. Vertical position keeps the same. 2. Horizontal position +1 pixel to new state if current position of cursor is not adjacent to the right border of user interface; vertical position keeps the same if current position of cursor is adjacent to the right border of user interface. 3. Interaction continues	Positive rewards proportional to reward density at next state from collected human user interface interaction data.
Stay	1. Horizontal position keeps the same. 2. Vertical position keeps the same 3. Interaction continuous	Positive rewards proportional to reward density at current state from collected human user interface interaction data.
Click	1. Horizontal position keeps the same. 2. Vertical position keeps the same 3. Interaction stops if last target button is clicked; otherwise interaction continuous	1. Positive rewards if acting on target button clickable zone in sequence. 2. Otherwise negative rewards.

Table 5-3: Agent's possible actions and its related state change and reward received.

From Table 5-3, it can be noticed that agent interaction with user interface only stops when clicking all target buttons in sequence. Otherwise agent keeps trying until success. One thing worth mentioned is action Stay. Action Stay is very special among one of the five navigation actions from two aspects:

1. It does not lead any change in states;
2. It can still gain rewards;

The existence of action Stay seems useless but it is a necessary component to Phase II evaluation model for two reasons.

1. In observing user interface interaction, it exists that cursor stays on a position.
2. Action Stay helps model to measure time regardless of reward distribution.

Phase II evaluation model is a discrete time model but is not measured real time. Measuring real interaction time is influenced by many factors such as cursor speed settings, DPI settings of mouse and screen resolution. Therefore, Phase II evaluation model uses action step as unit to measure time. Each step of action represents for a unit of time. Like a ticker-tape timer, Agent keeps making action with constant interval of 'one step' making. If printing a dot on cursor position of user interface after each action, connecting these dots generates the path of interaction. The speed of cursor at a position is inverse proportional to the density of dots at the position.

+	+	+	+	+
+	-	-	-	+
B+	-	A:++	-	+
+	-	-	-	+
+	+	+	+	+

Figure 5-7: Case without action Stay might lead different interaction results. Sign in each cell represents for reward.

Without action Stay, it might happen that cursor oscillates between high reward current position and low reward adjacent position. This oscillation can also be used to approximate steps spending on one position but it influences and complicates reward settings on whole user interface. For example, a high positive reward position is surrounded by negative reward position as shown in Figure 5-7. Each cell represents a pixel on user interface. Sign in each represents for positive or negative reward. If agent has no option to stay, the reward to enter high reward position A becomes much lower since reduced reward by adjacent negative reward and discounted future rewards. Another concern is that if action Stay can keep gaining reward at the same position cursor can be trapped in a high reward position. Notice discounting factor λ is less than 1. The total rewards from non-target area is bounded by:

$$\sum_i \lambda^{i-1} r^i \leq \frac{max\{r_i\}}{1-\lambda} \tag{5-7}$$

Therefore, if target reward is properly set, cursor will not be trapped before task completion. More details about setting up rewards will be discussed in discussion of reward function.

Reward function

Reward function plays a very important role in reinforcement learning. Generally, reward function describes how agent should act or what are expected behaviors of agent. In Phase II evaluation model, two behaviors are praised: 1) Action that complete interaction task. 2) Action that mimics human's behavior. Action that complete interaction task is straight to define. As long as agent chooses action Click action within the area of desired button, agent receives rewards. The problem is how to set proper rewards for agent acting in a similar way to humans. Question then becomes what can be regarded as similar to human's behavior on a user interface interaction. Human user interface interaction has both properties of uncertainty and trend. On one side,

human user interface interaction cannot be expressed using a function or a curve. There exist individual differences in interaction habits. There also exists randomization within individual's human interaction. The same interaction task performed by the same human user cannot obtained exactly the same interaction result. Therefore, user interface interaction has randomization process. On the side, human users can always complete task even though every time their interaction traces are not the same. It seems that there exists a trend to task completion.

Phase II evaluation model has non-stochastic settings for target button reward function and stochastic settings for human behavior reward function. One thing to clarify is that interaction result is different with reward function. Reward function refers to desired or expected position that interaction might pass. Interaction result refers how agent or user actually interacts with user interface and may not pass all the positive reward positions. If discounting factor $\lambda = 1$, interaction results pass all positive reward positions. If factor $\lambda < 1$, interaction results may sacrifice present reward in exchange of long term rewards. To build stochastic reward functions for human behavior, human data of interaction result is required. However, to collect user interface interaction results under different states of cursor and different types of user interfaces is impossible. Required data size is too large to collect. A user interface of size 400 by 300 pixels requires $400 \cdot 300 \cdot 256^{400 \cdot 300}$interaction data to cover human interaction under different conditions at least once. Therefore, it cannot be avoided to approximate undiscovered interaction states and model of environments' interaction results.

One way to approximate undiscovered interaction states and model of environments' interaction result is to use data from the most similar task under a similar user interface and implement Monte Carlo method to find reward function. Similarities of tasks are measured using initial cursor position and the center of the target button position. Similarity of user interfaces is

determined by pixel matrix cosine value I. Suppose an interaction task is defined by initial position P and end position Q. Matrix cosine is I. An undiscovered interaction states and model of environments' interaction result can be approximated with equation 5-8 and 5-9.

$$T = \text{argmin}_{T_i} \left| ||P - P_i|| + ||Q - Q_i|| \right| \tag{5-8}$$

$$U = \text{argmin}_{U_k} I = \text{argmin}_{U_k} \frac{\left(\sum_{i,j} U_{ij} \cdot U_{k_{ij}} \right)^2}{\left(\sum_{i,j} U_{ij}^2 \right) \times \left(\sum_{i,j} U_{k_{ij}}^2 \right)} \tag{5-9}$$

If T and U have at least one instant of intersection, intersection can be directly used as interaction result to set up reward function. This solution can generate accurate results under the condition of sufficient training data cover all conditions' user interface interaction result but is also vulnerable to condition that has not been met. If the absolute value of $\min_{T_i} \left| ||P - P_i|| + ||Q - Q_i|| \right|$ or $\min_{U_k} \frac{\left(\sum_{i,j} U_{ij} \cdot U_{k_{ij}} \right)^2}{\left(\sum_{i,j} U_{ij}^2 \right) \times \left(\sum_{i,j} U_{k_{ij}}^2 \right)}$ is large, interaction results cannot provide useful guide to set up reward function or even misleading.

The other way is to involve all the collected data of user interaction results to approximate undiscovered interaction states and model of environments' interaction result. The problem of previous solution is that the most similar task and user interface might have bias which may mislead agent's behavior. If user interface interaction can be approximated based on average of all interaction results, unsymmetrical effect or biased of collected interaction results can be canceled internally. This method does not require sufficient data to cover interaction results of all conditions and only requires symmetric sampling during data collection. Undiscovered interaction result D is expressed as combination of collected interaction results D_i as shown in equation 5-10.

$$D = \sum_i \beta_i D_i \tag{5-10}$$

$$D_i = I_i \cdot U_i \tag{5-11}$$

Parameter β_i represent for contribution of collected interaction result D_i and are related to similarity of user interfaces and interaction tasks.

$$\beta_i \propto I \tag{5-12}$$

$$\beta_i \propto 1 - \frac{|P^x - P_i^x|}{W} \text{ and } \beta_i \propto 1 - \frac{|P^y - P_i^y|}{H} \tag{5-13}$$

$$\beta_i \propto 1 - \frac{|Q^x - Q_i^x|}{W} \text{ and } \beta_i \propto 1 - \frac{|Q^y - Q_i^y|}{H} \tag{5-14}$$

Besides, discounting factor λ can reduce the effect of misleading human interaction reward since the large $\big||P - P_i|\big| + \big||Q - Q_i|\big|$ value increases cost to obtain human interaction rewards.

For non-stochastic settings for target button reward, it requires properly setting up rewards for target button rewards to avoid failing to complete task since Phase II evaluation model does not model error. Suppose human interaction rewards are bounded by r and target button reward is R_{target}. Then the total reward received from human interaction rewards is bounded by:

$$\sum_i r_1 + \lambda r_2 + \lambda^2 r_3 + \cdots \leq \sum_i r + \lambda r + \lambda^2 r \cdots = \frac{r}{1-\lambda} \tag{5-15}$$

The maximization reward from target button is:

$$\lambda^{\big||P-Q|\big|_1} R_{target} \tag{5-16}$$

To guarantee the task completion, it needs to have:

$$\lambda^{\big||P_1-P_2|\big|_1 + \big||Q_1-Q_2|\big|_1} R_{target} \geq \frac{r}{1-\lambda} \Longrightarrow R_{target} \geq \frac{r}{\lambda^{\big||P_1-P_2|\big|_1 + \big||Q_1-Q_2|\big|_1}(1-\lambda)} \tag{5-17}$$

Since target button reward is R_{target} has highest positive reward, then the total rewards obtained by agent is bounded by:

$$R_{total} \leq \frac{1}{1-\lambda} \cdot R_{tagret} \tag{5-18}$$

which means if all reward value $r, R_{target} < \infty$ and $\lambda \in (0,1)$ total rewards always converge.

Example

An example is provided to introduce how reward function is obtained. Suppose reward function for user interface A performing navigation task from Button A to Button B is needed as shown in

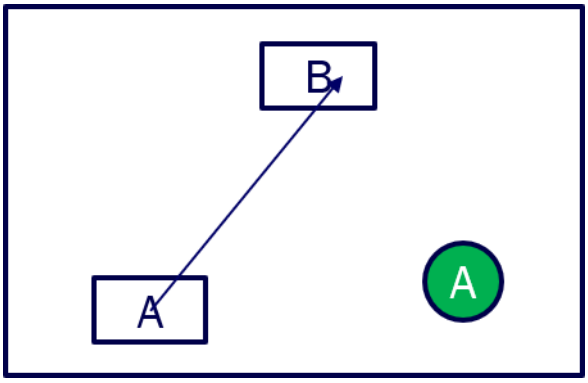


Figure 5-8: User interface which needs to be assigned with reward function

Three sets of training data B, C and D are implemented to provide reward function for user interface A and its task as shown from Figure 5-9 to Figure 5-11.

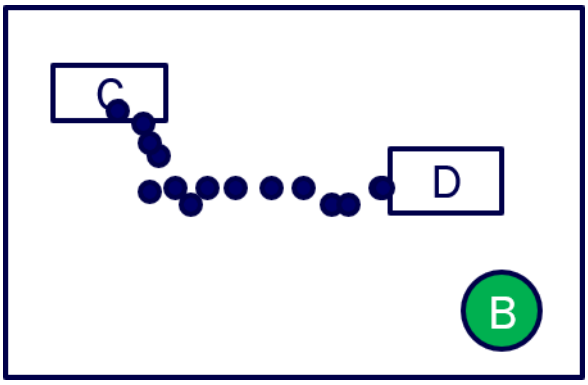


Figure 5-9: Training data B

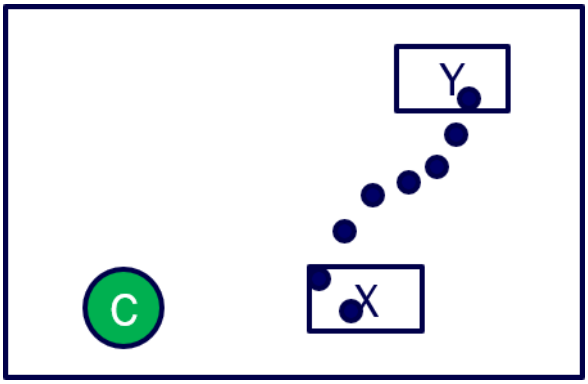


Figure 5-10: Training data C

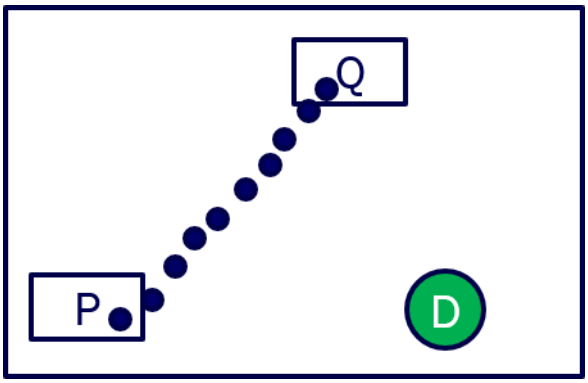


Figure 5-11: Training data C

After compare three sets of training data with user interface A, training data of B, C and D have different user interface similarities and task similarities with user interface A its task. Therefore, they have different contribution to user interface A and its task.

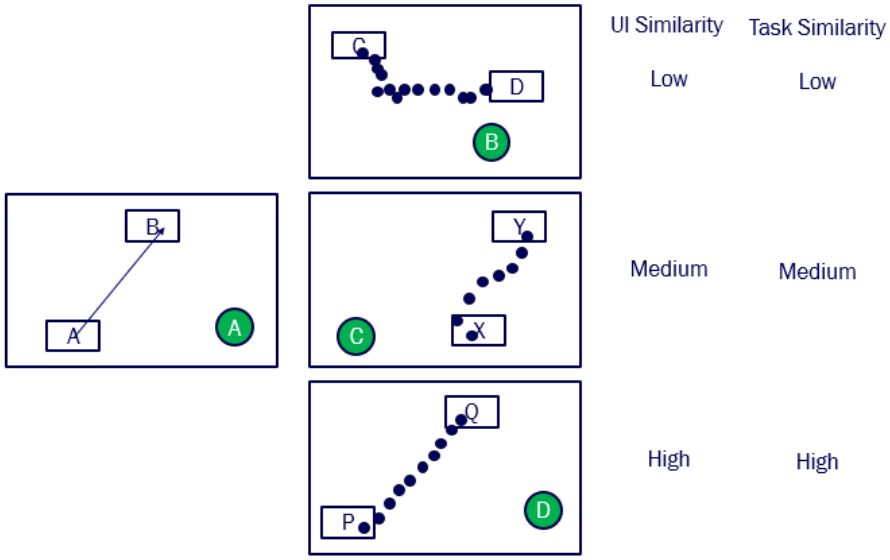


Figure 5-12: Comparison to training data

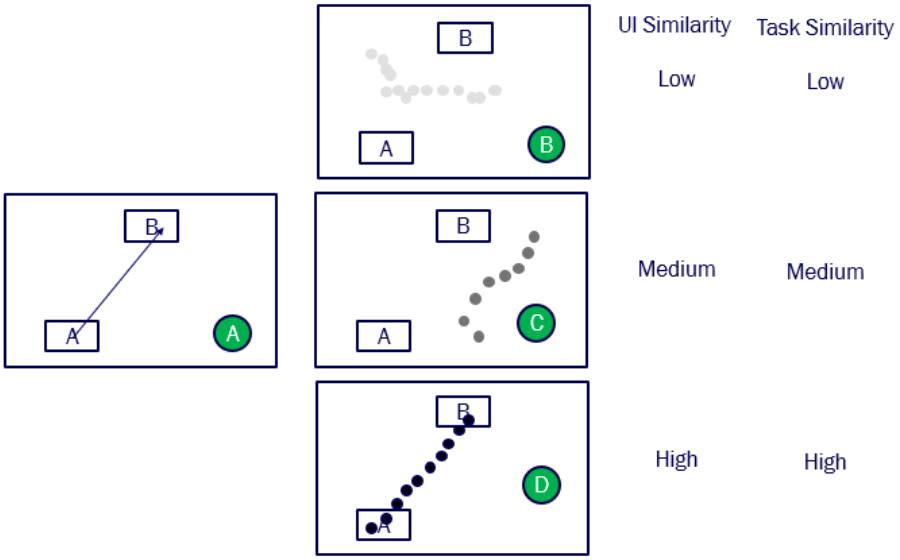


Figure 5-13: Contribution to user interface A and its task

The contributions from training data B, C and D are based on equations from 5-9 to 5-14.

Value function

In the prototype mentioned previously, Q learning method is used to implement model and Q table is the core of Q learning method. Q table is a reference dictionary recording estimation of each action under different states. During interaction with model of environment, Q table is updated through approximating $Q^{\pi}(s_{t+1}, a)$ as $max_a Q(s_{t+1}, a)$ and moving towards true value by learning rate α. Q table is basically a mapping from (s, a) to action value $Q^{\pi}(s, a)$ under policy π. Deep Q network replaces Q table with neural network to generate Q value. Neural network has structure of nodes and layers. A neural network has an input layer, an output layer and hidden layers. Each layer consists of nodes. Each node connects with every node in next layer. Node value is equal to linear combination of previous layer with bias going through activation function. In general, neural network can be regarded as a function estimator based on inputs and outputs as shown in equation 5-19.

$$(s, a) \Longrightarrow \text{Neural network: } f(\cdot) \Longrightarrow Q(s, a) \tag{5-19}$$

Action value can be then represented by:

$$Action\ Value\ = Q(s, a) = f(s, a, \omega) \tag{5-20}$$

ω represents for parameters of neural network. State s is a of high dimensional in comparison with action a. To keep the neural network neat, input layer of neural network only includes states s and output layer of neural network generates a vector representing for values of actions. Then the neural network of Q value can be shown in equation 5-21.

$$Q(s) = f(s, \omega) \tag{5-21}$$

Using neural network rather than Q table to generate Q values has several benefits. First, neural network has better generality than Q table. Q table do not update Q table for state that has not been met before. When encountering new state, Q learning selects and executes action based on its initial value. Neural network can produce Q value based on parameter ω therefore has better chance to make better option. Second, neural network is neat and consistent in structure. When the total number of states have exponential growth, Q table becomes a very long 2D table and is very difficult to build and update. Third, estimating Q value with neural network has good connection with convolution neural network for user interface evaluation of high dimensional sensory input. One challenge of neural network is model selection determining how many layers of neural network and how many nodes each. There is no clear rule corresponding to structure of neural network for specific type of problem. Some tests are necessary in training neural network for better results. In Phase II evaluation model, we use neural network with two hidden layers. Each layer has 10 nodes. To solve for parameters of neural network, RMS propagation method is implemented to minimize loss function. Similar to Q learning method, difference between target Q value and current value can be used as loss function.

$$L(\omega) = \mathbf{E}\left[\left(r + \lambda \max_{a'} Q^{\pi}(s', a', \omega) - Q^{\pi}(s, a, \omega)\right)^2\right] \quad (5\text{-}22)$$

Deep Q network uses experience replay updating parameters to get over data correlation and non-stationary distribution. While performing mini-batch stochastic gradient descent optimization, it assumes independence between sample data so that noise of data cancels within mini-batch. Neural network model as a supervised machine learning method also requires data with independent distribution. In reinforcement learning method sample data is correlated. While agent interacting with user interface, it stores (s_t, a_t, r_t, s_{t+1}) into mini-batch and randomly choose data from mini-batch of 5 to train.

5.3 Implementation Methods

5.3.1 Data collection

30 participants of 18 to 26 years old University of Michigan students were recruited through emails in three batches to participate in one hour user interface evaluation study. Participants are from group of normal novice users. Recruited participants are required to have three years' experience of using any user interface on computer based information system. Participants should not have participated the research before or have any usability evaluation experience. Collected data was divided into two groups, 25 subjects' data were used as training data for interaction simulation and 5 subjects' data were used as verification data. Each subject was told to perform a user interface interaction task on a prompted user interface based on instruction on the screen. When experiment started, task instruction was shown on the bottom of the experiment window and an empty user interface with a free move cursor was shown on the top of the experiment window. When subject was ready, he/she left clicked mouse anywhere within

the window to start. Every 20±1 milliseconds, cursor position was recorded as human interaction data. After subject successfully completed task, this trial of experiment ended. New trail of experiment showed up and repeated. Data collection user interface is as shown in Figure 5-14.

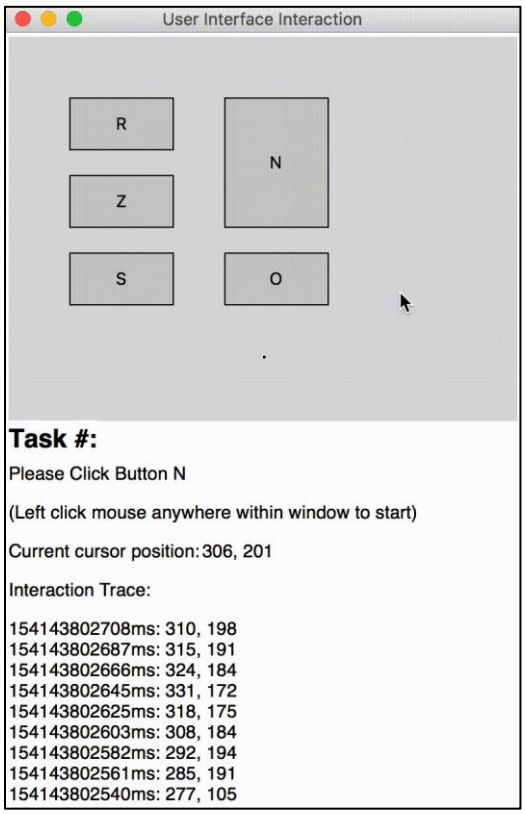


Figure 5-14: Phase II data collection user interface.

5.3.2 Model Training

Deep Q net work is implemented to train agent's behavior. With assigned reward function on user interface and defined six possible actions, agent keeps interacting with user interface gaining feedback to update parameters of neural network in order to achieve maximum of total expected accumulative rewards.

5.3.3 Evaluation Quantity

Four components mentioned above work together and generate simulated user interface interaction results. The remaining question is how to properly use generated interaction results to evaluate a user interface and makes adjustment to it to improve its usability. There does not exist a quantity or metric describe user interface usability from interaction results. Therefore, during data analysis we brainstormed following quantities to evaluate some aspects of usability.

1. Number of learning episodes: Agent needs to learn how to interact with a user when it meets a user interface for first time. From initially random lattice walk to finally interact as expected, it goes through a long learning process. The length of learning process can to some degree reflect the learning difficulty to use interface.
2. Interaction steps (Interaction time): Interaction steps is not a measure of total travel distance but a measure of time like a ticker-tape timer. On the same dimension user interface, interaction steps can reflect efficiency to complete interaction tasks.
3. Proportion of action Stay: Action Stay is special among five navigation actions as mentioned earlier. Stay action is selected at high reward position. If the proportion of Stay is large, agent stays much during interaction which shows users have hesitation during interaction.

Chapter 6 Phase II Results and Discussions

Chapter Summary

This Chapter presents the results of Phase II evaluation model. It first presents the simulated user interface evaluation result including their learning process and interaction performance. Then a verification study of simulated interaction results using human data of interaction is conducted. After verification, quantity analysis and user interface suggestions are discussed. Lastly, it mentions the benefits and limitations Phase II evaluation model.

6.1 Simulated Interaction Results

The generated model is tested on different user interfaces. Table 6-1 shows the parameters used in Phase II evaluation model.

Learning rate: α	Reward decay: λ	$\epsilon - greedy$	Memory size	Batch Size
0.01	0.9	0.9	500	32

Table 6-1: Parameters for simulated user interface interaction results.

Figure 6-1 shows the structure of Phase II evaluation model. Phase II evaluation training on average 4000 episodes. It estimates 120000 states' value under 6 actions. Trained neural network can work well in estimating these 720000 values. Network structure of Phase II evaluation model is shown in Figure 6-1.

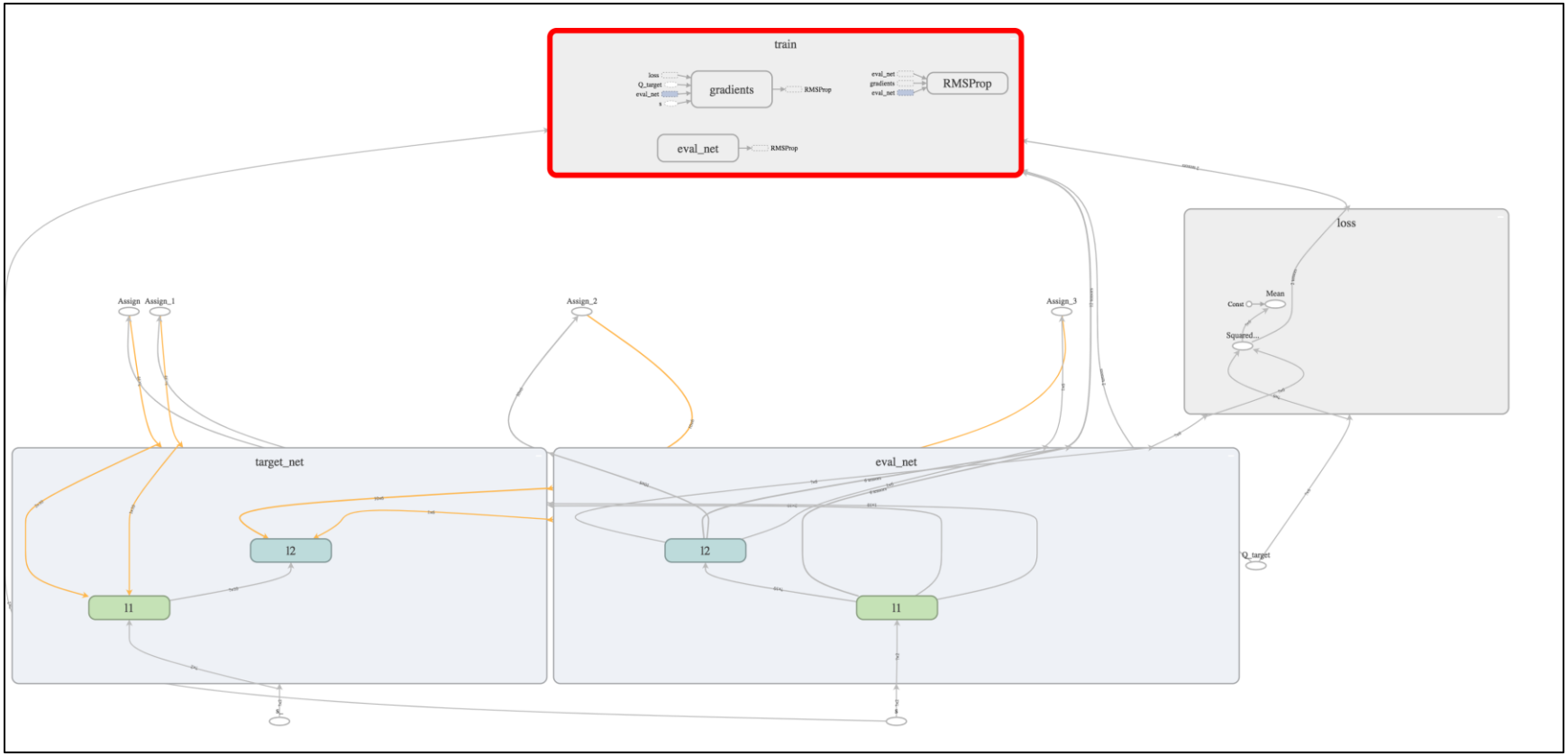


Figure 6-1: Phase II evaluation model structure generated by tensorflow board.

User interface consisting of three buttons, T, J and W is used as an example to demonstrate simulation of user interface interaction. Task is to navigate to button T and click button T. The initial position of cursor is set to be in the middle of the user interface. User interface is designed to test whether interaction simulation is able to complete task under a simple condition. Figures 6-2 to 6-5 show agent's interaction with interface during different periods of learning process.

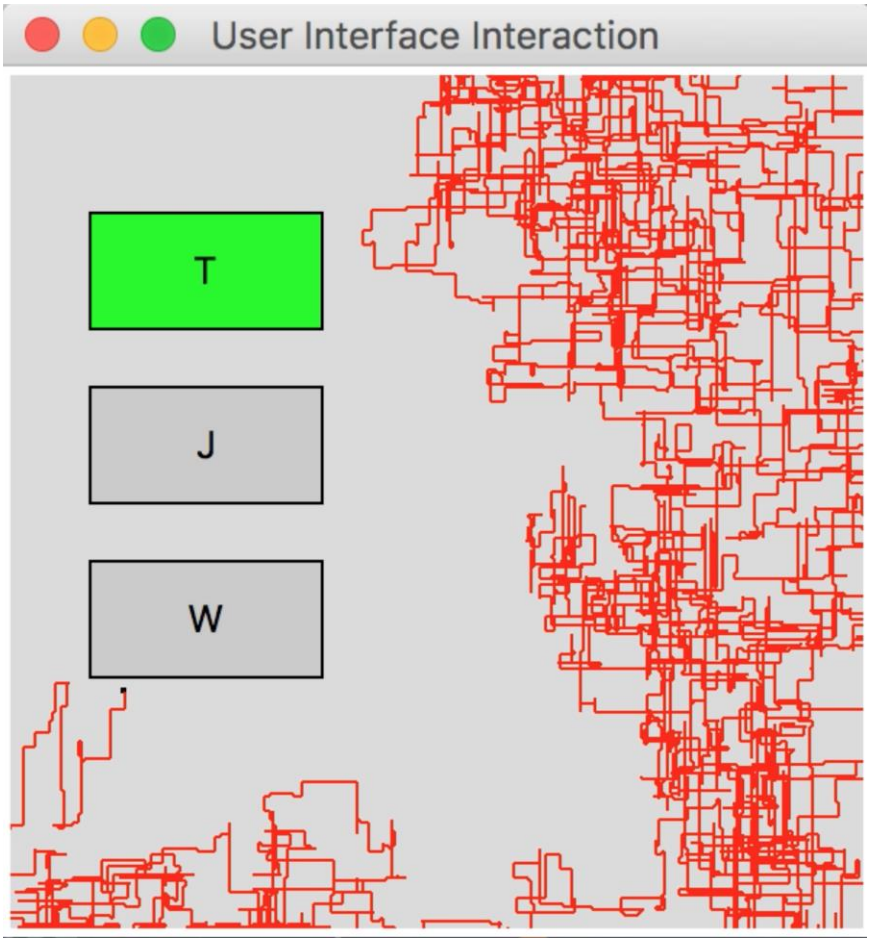


Figure 6-2: Interaction training process after 100 episodes

Figure 6-2 shows the training process after 100 episodes training. It can be noticed that agent is exploring states like random lattice walk. At this stage, agent cannot give good estimates to the true value of each state under different actions.

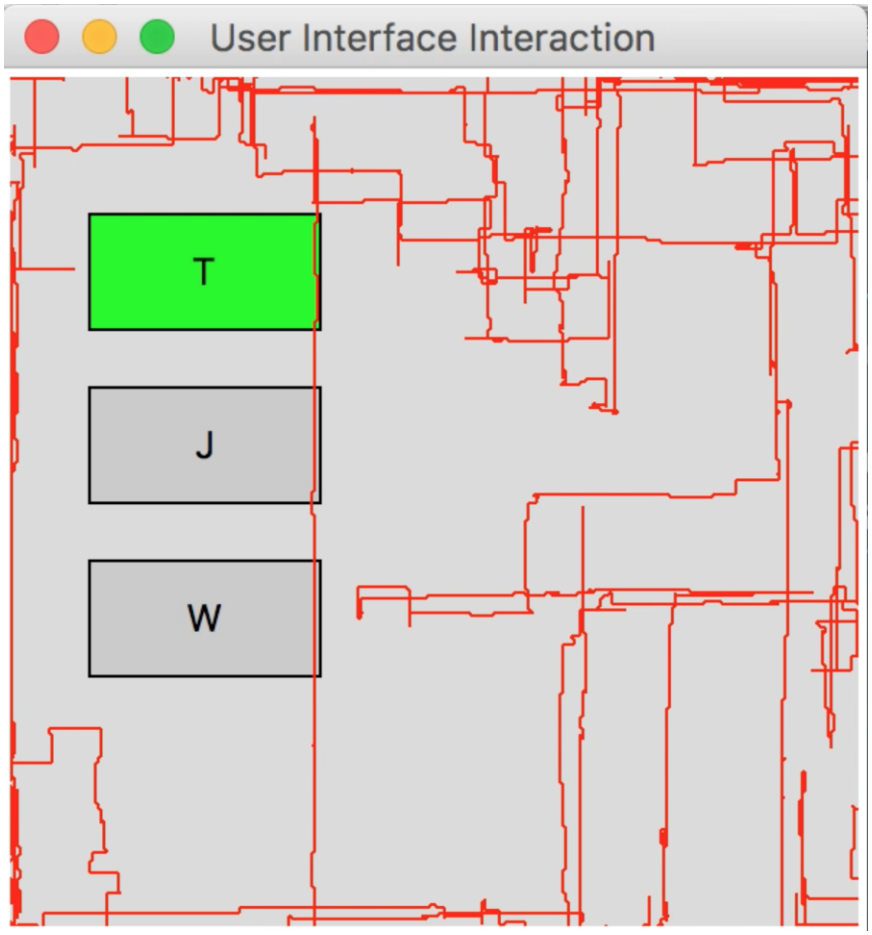


Figure 6-3: Interaction training process after 500 episodes

With Agent updated neural network for Q value, agent behavior becomes more effective. Before not all the state value is well estimated, exploration behavior may lead to large deviation to task completion. $\epsilon = 0.9$ means every 10 steps of actions there will be a random action. This random action selection does not follow the Q value generated by neural network to make decisions. The advantage of using $\epsilon - greedy$ policy is to increase the chance to find better policy to complete tasks and mimic of human behaviors.

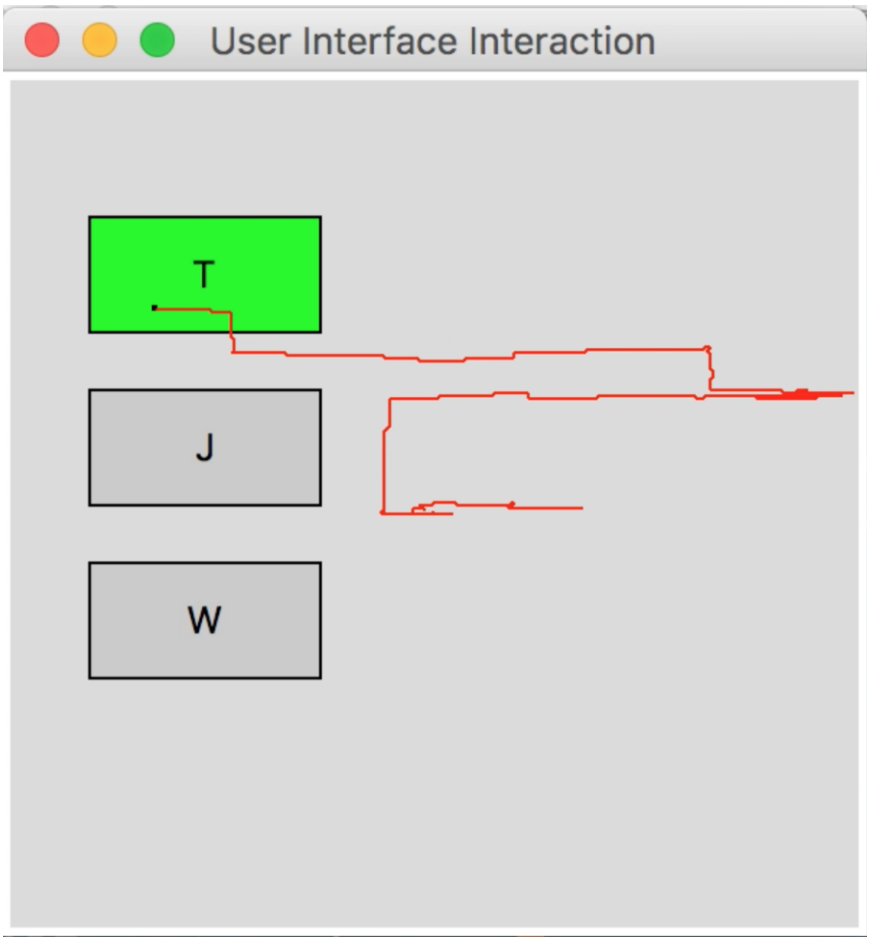


Figure 6-4: Interaction training process after 1000 episodes

After 1000 episodes' training, some neat and efficient interaction starts to show up often. It indicates that estimated Q value is close to the true value. From this time point, the training progress starts to become slow since parameter update becomes slow. Sometimes, it could happen that agent traps at suboptimal solution during this period of training.

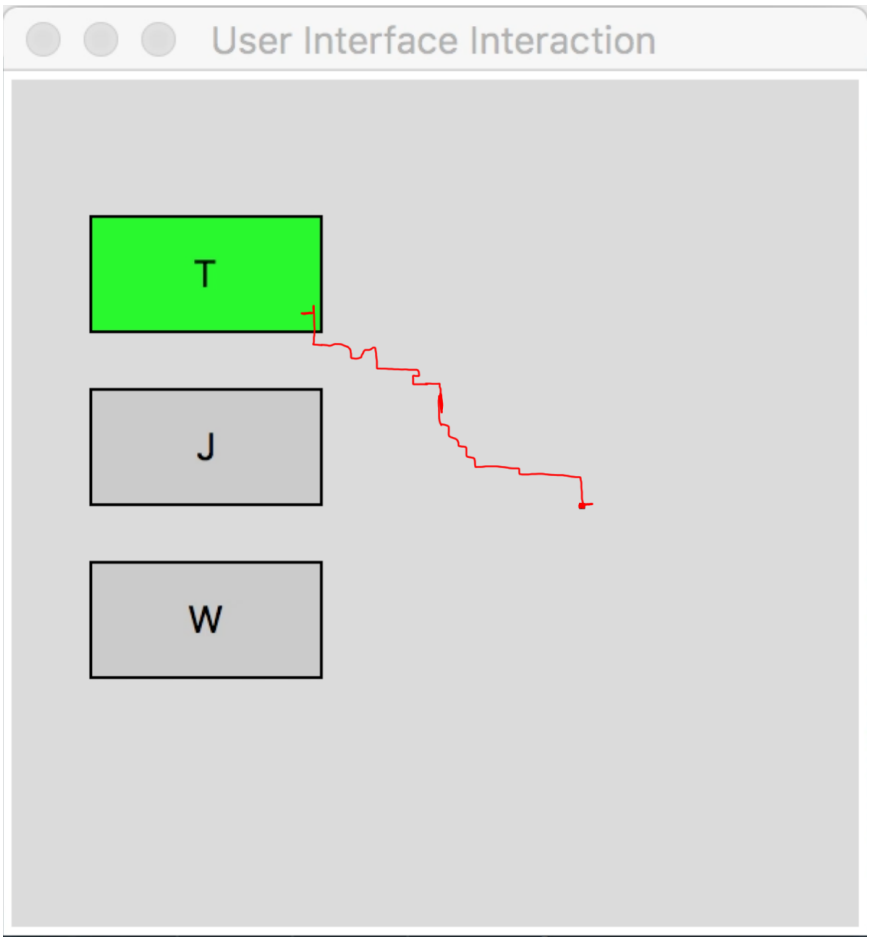


Figure 6-5: Interaction training process after 4500 episodes

After 4500 episodes' training, the agent is able to interact with user interface in a human understandable way. It indicates that neural network can almost estimate Q value of all states well. Further training might improve performance. From Figure 6-2 to Figure 6-5, agent tends to navigate more on the right side of the user interface. In Figure 6-5, agent does not go straight to target button. There usually exists two possibilities: 1) Agent is trapped in a suboptimal position 2) Human behavior rewards distributed more on the right side of user interface than that of left side. User interface is divided by 2 by 2 squares. The average human behavior rewards are:

0.89	**0.88**
0.68	**0.74**

From human interaction rewards distribution, human users navigating with this user interface tend to be on the upper area of user interface. This phenomenon is reasonable and can be explained with Monte Carlo method. In implementation method, we have mentioned that coefficient of user interaction distribution is related to similarity of the task. Dashed rectangle is the most influential area to user interaction results. From the graph, it can be noticed that upper right area within rectangle is much larger than that in lower left area within rectangles.

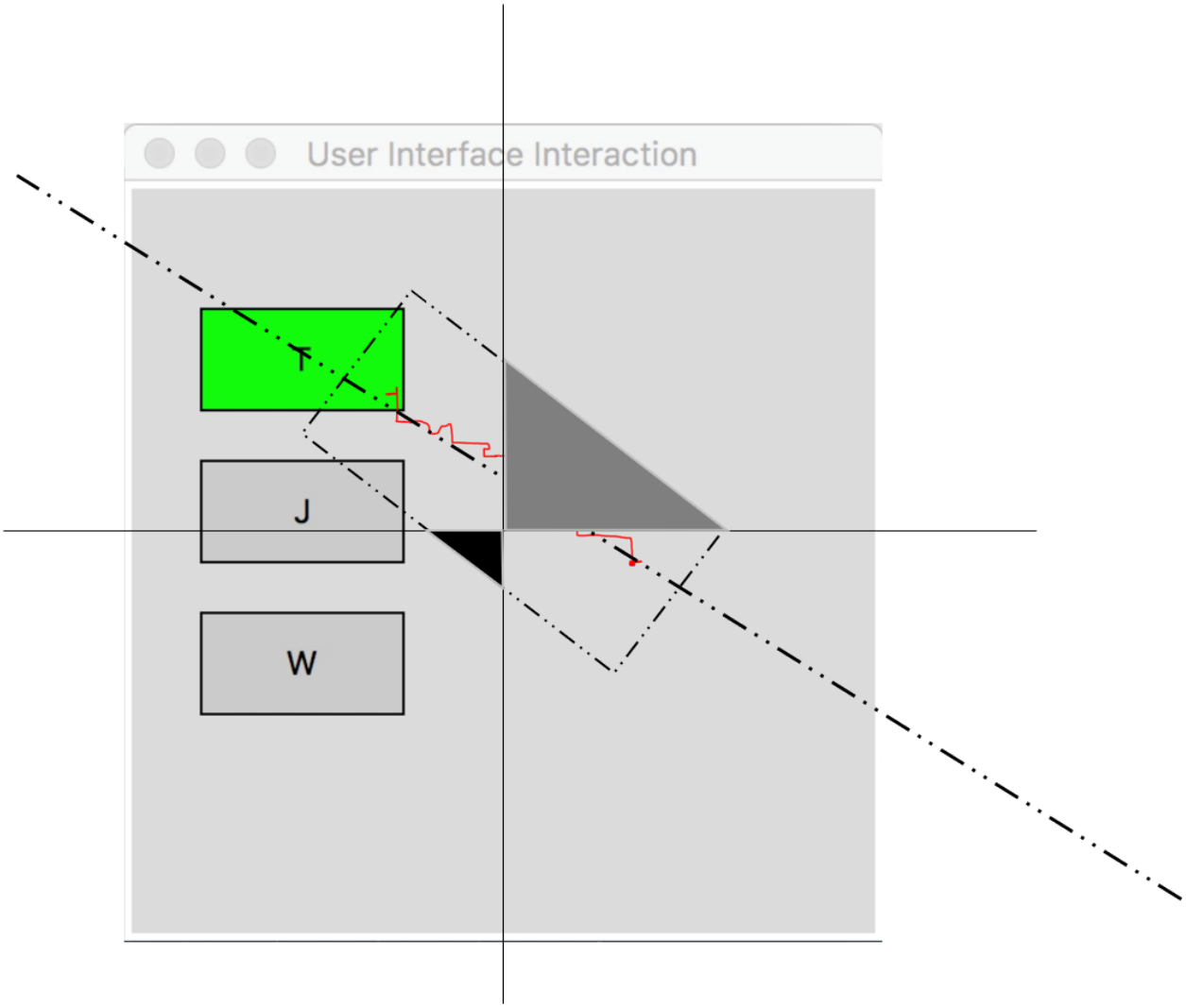


Figure 6-6: Upper right area within rectangle is much larger than that in lower left area within rectangles. Based on Monto Carlo method, the probability that influential human behavior reward falls in upper right area is larger than that in lower left area.

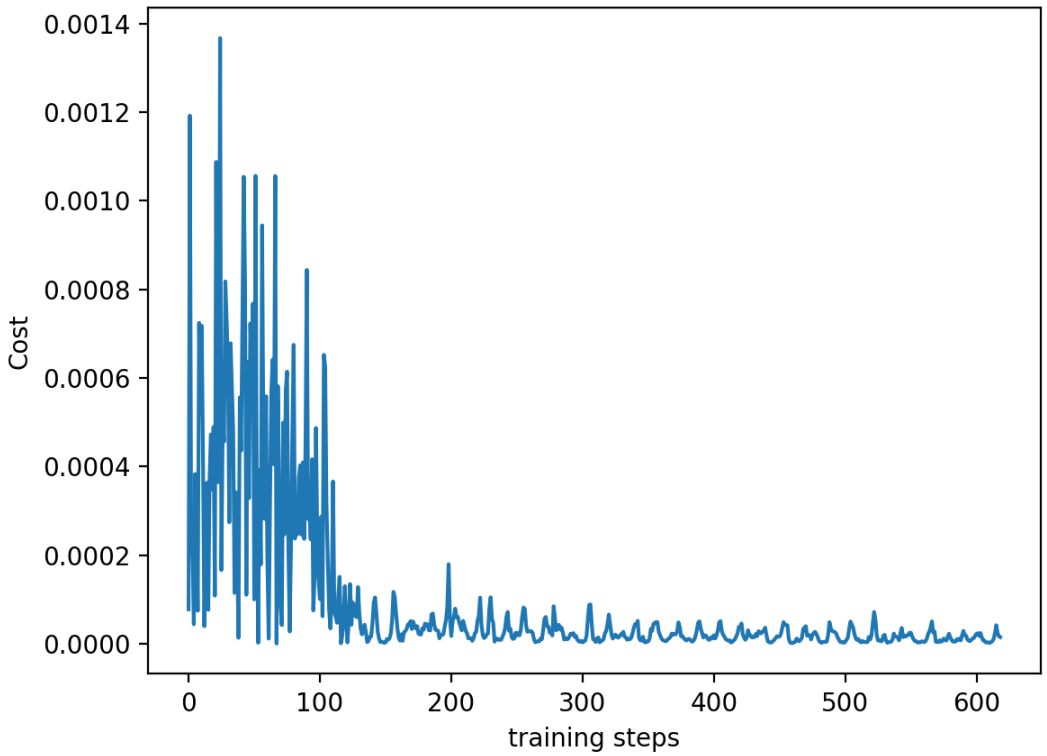


Figure 6-7: Loss function gradually converges at first 600 steps while neural network updates parameters

Figure 6-7 shows the relationship between cost of loss function and training steps. Before 100 steps, loss function fluctuates which means neural network cannot correctly estimate true state values. After 100 states, loss function gradually converges which means current state value estimation is consistent with environment feedback and parameter adjustment.

Figure 6-8 shows loss function in a long run. Before 5000 steps, neural network keeps low cost of loss function which means it already updates parameters and successfully estimates all state values that it has met. At around 5000 steps, there is an abrupt increase of loss function. Abrupt increase means estimated state value has huge difference with actual findings. Neural network needs to readjust parameters to correctly estimate each state value under different

actions. This usually happens when agent explores actions under new state. It is helpful for agent to jump out of trap of local maxima and search for better policies.

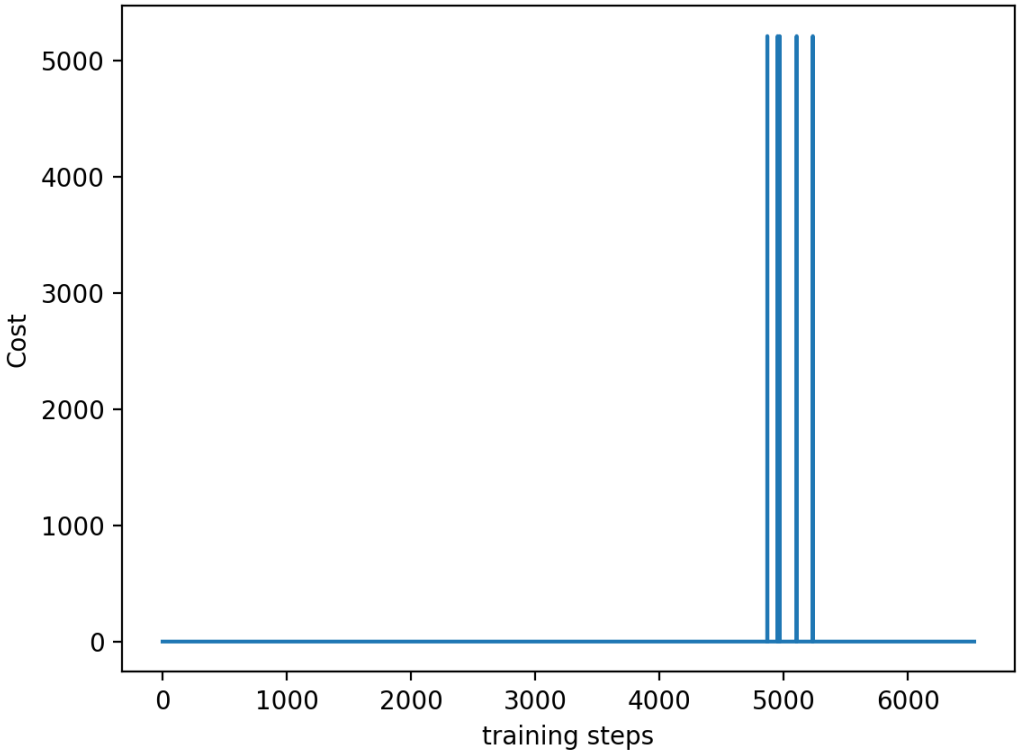


Figure 6-8: Abrupt increase in loss function refers to exploration of new policies and neural network needs to be updated its parameters.

6.2 Model Verification Study

After modeling the interaction simulation, I conducted a verification study between simulated user interface interaction results and collected human interaction results. Five participants' user interface interaction data are used to perform this verification study. I tested 10 user interface tasks. All these tasks are from human interaction results of five participants in verification group. Agent is asked to perform exactly the same ten task as participants. Since agent simulated interaction and collected data of human interaction perform the same task under the same user interface environment, they are expected to have similar interaction results. A problem is what mathematical model can be used to measure the similarity between user interaction results. Variation analysis is usually used to analyze distribution of data but when used to measure the similarity of user interface interaction this method does not work well without assuming some correct interaction pattern. Variation within human data of interaction is large. Therefore variation does not always contain necessary information to verify interaction results. The original intention to use reinforcement learning method rather than other supervised machine learning method to research on interaction is interaction pattern is unknow. The difficulty to verify whether a simulated interaction result is similar to human interaction results is that human interaction results themselves are stochastic process. In the probability view of interaction results, no interaction results can be regarded as wrong. There is only interaction results with low probability. Therefore, we use envelop of human interaction results to verify the simulated interaction results. Envelop of human interaction results can be regarded as a high probability area in which interaction result may show like electron cloud. Envelop refers to a curve tangent to each of a family of curves. Simulated interaction results are usually not

mathematically smooth and not enough data can be obtained to support finding mathematical envelop of interaction results. Thus, the combination of parts of interaction results that enclosed maximum of area is approximately regarded as envelop of interaction results. Therefore, verification study is conducted through visually checking whether simulated interaction results fully, partially or not in envelop of interaction results. Interaction results within envelop means there is high probability that statistically they are reliable and can be treated as human interaction results. It cannot be used as criteria to judge whether a curve is human interaction results.

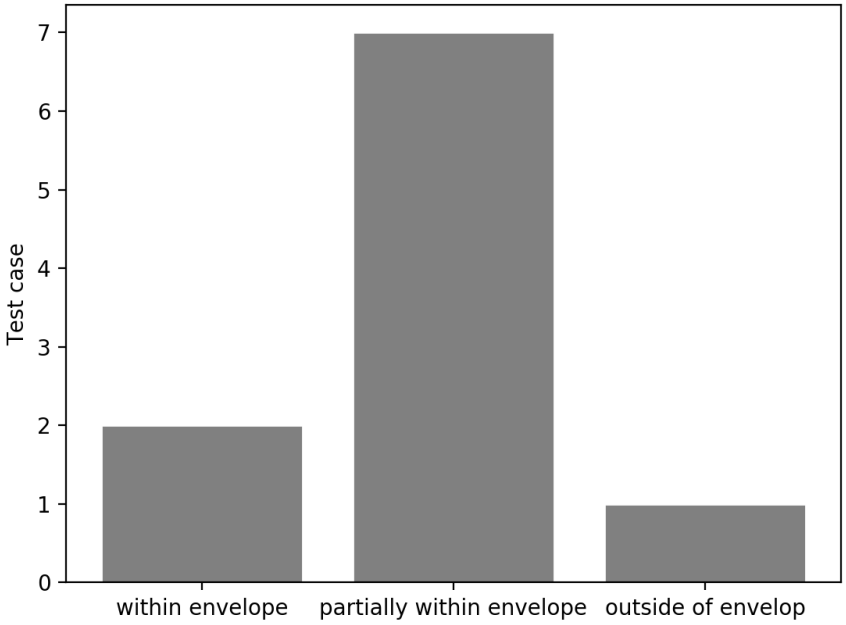


Figure 6-9: Number of tasks that simulated interaction results are within the envelop of collected interaction results.

Figure 6-9 shows the results of verification studies. Results of two test cases are fully within the area of envelops. Results of seven test cases are partially within the area of envelops. One result of test case is outside of envelop.

Five different participants perform ten one-step interaction tasks under ten user interfaces. Green button is the target button to complete task and during experiment participants only see buttons with color of gray scale. Initial position of cursor is randomly set within user interface. Task is to navigate from current position cursor to target button area and click button. Black curves on each user interface refer to collected human interaction results. Red curve on each user interface represents for the simulated interaction results. Since in Phase II evaluation error is not included, agent keeps interacting with user interface until successfully complete tasks. In the 10 test cases, agents' of interaction are trained around 1000 to 2000 episodes. Verification result is visualized in Figure 10 to Figure 19.

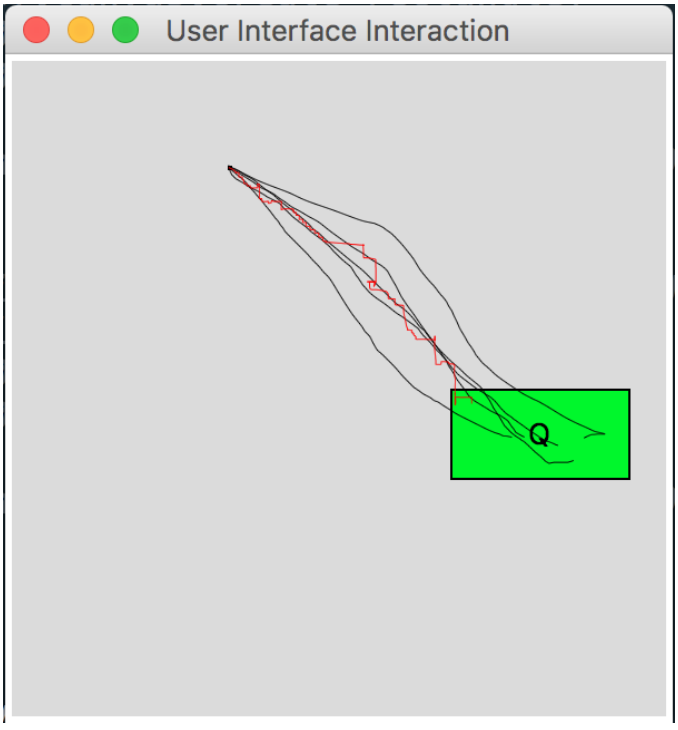


Figure 6-10: Test Case 1

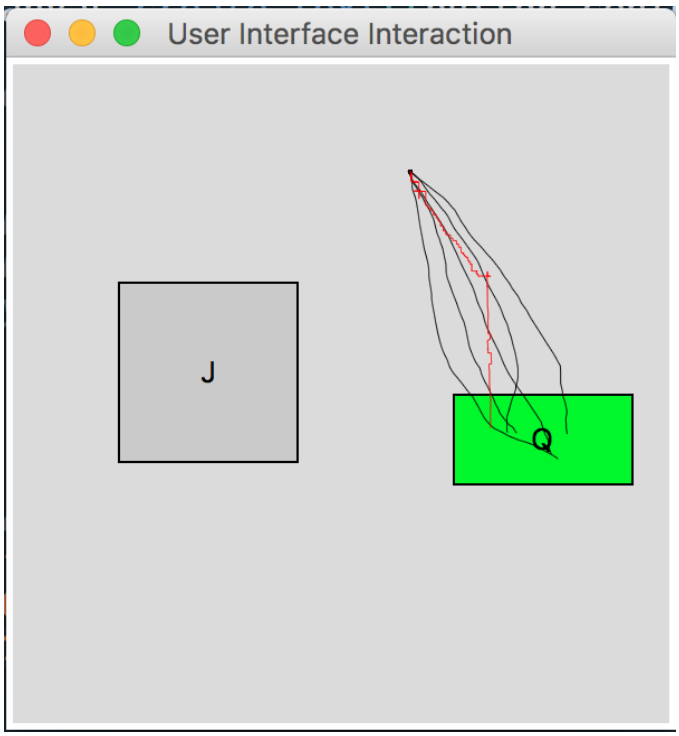


Figure 6-11: Test Case 2

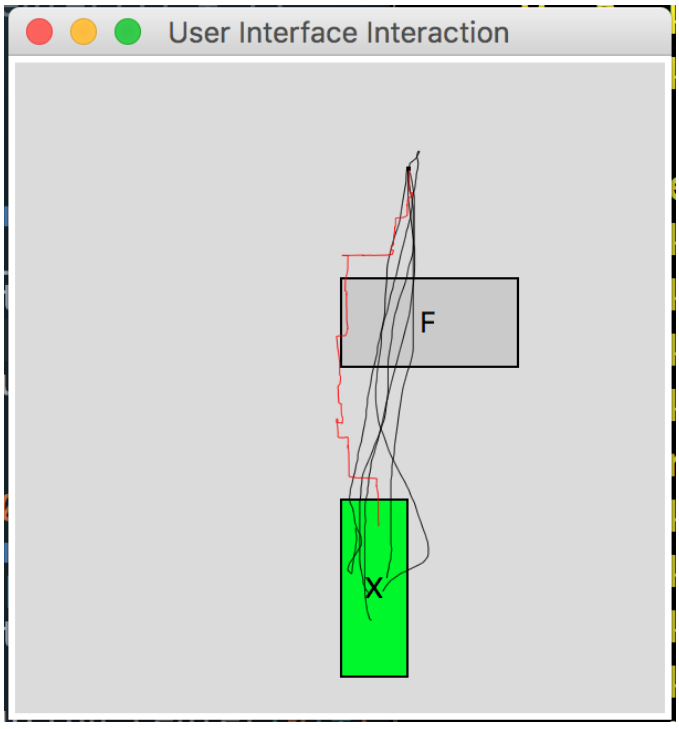


Figure 6-12: Test Case 3

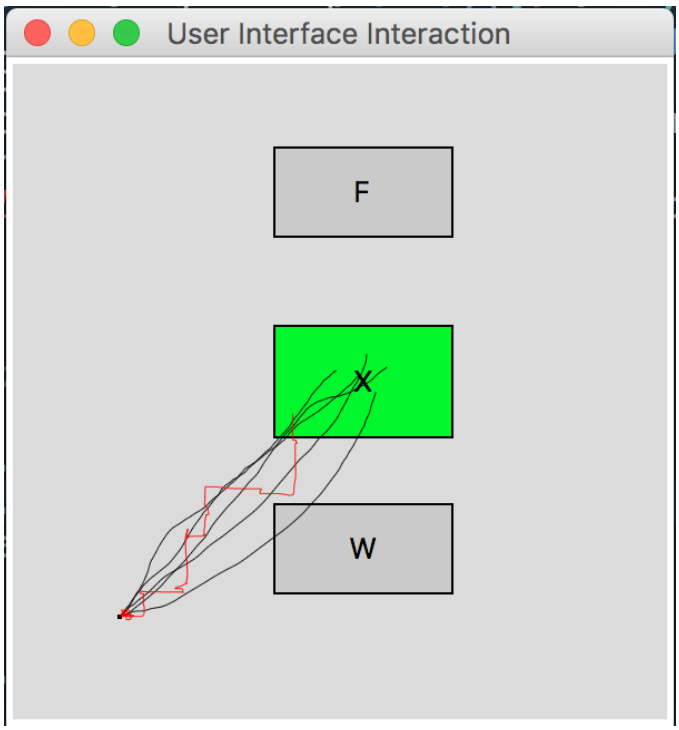


Figure 6-13: Test Case 4

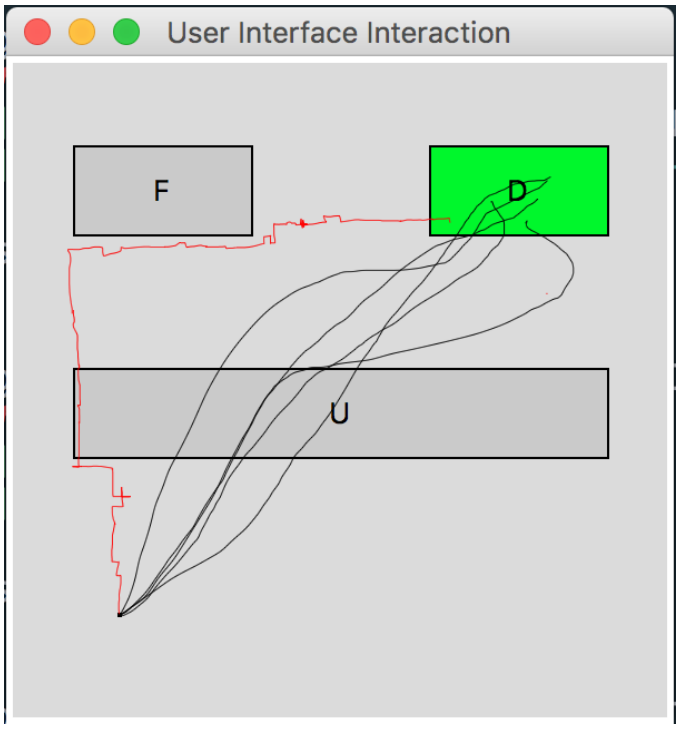


Figure 6-14: Test Case 5

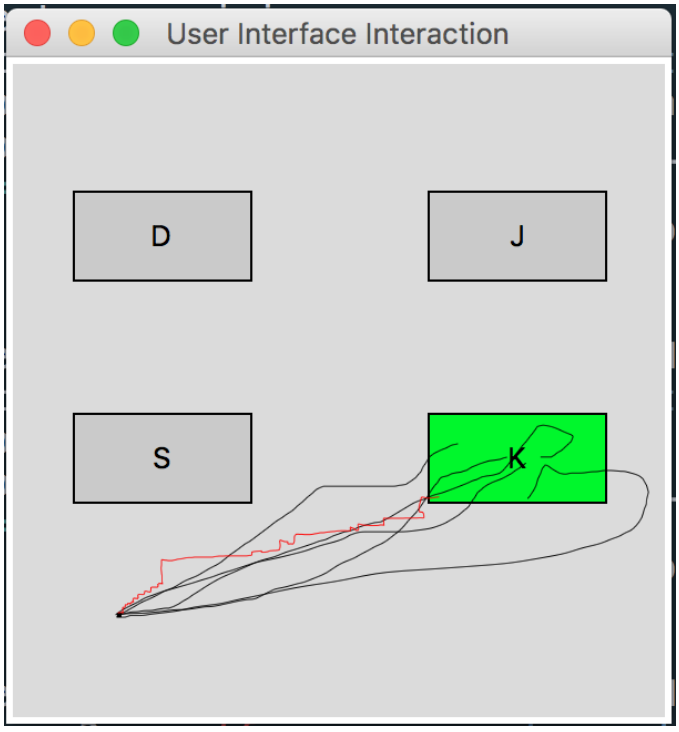


Figure 6-15: Test Case 6

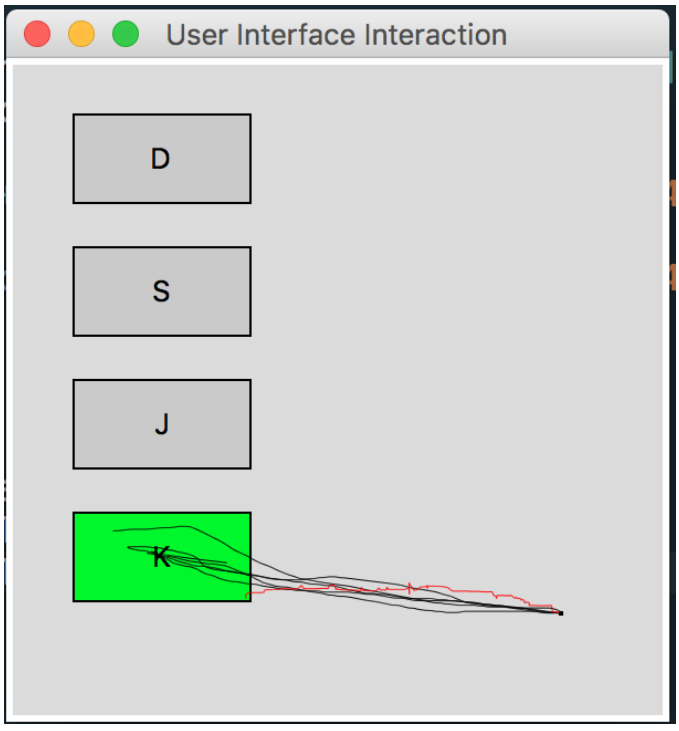


Figure 6-16: Test Case 7

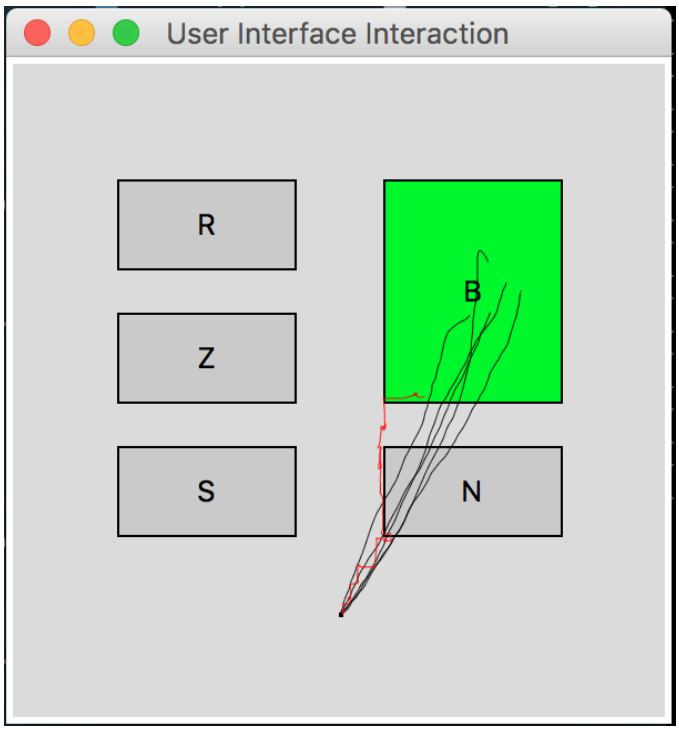


Figure 6-17: Test Case 8

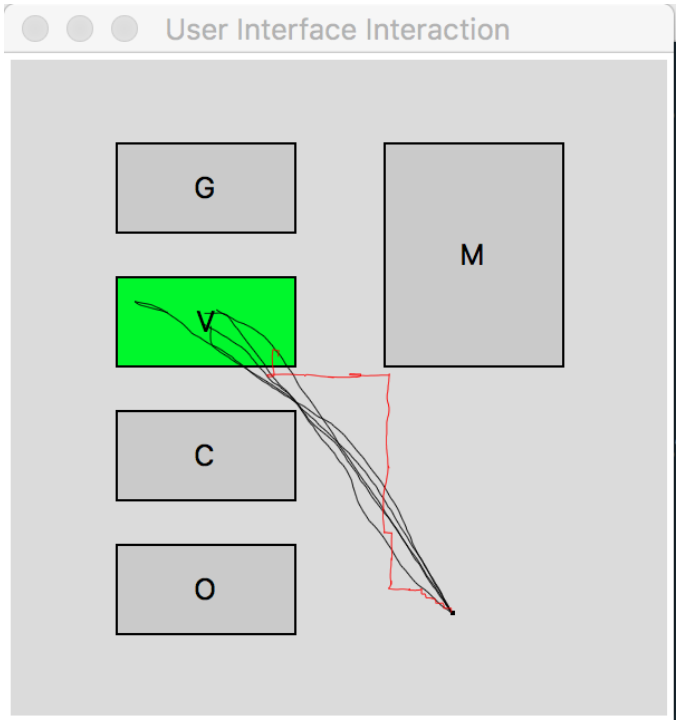


Figure 6-18: Test Case 9

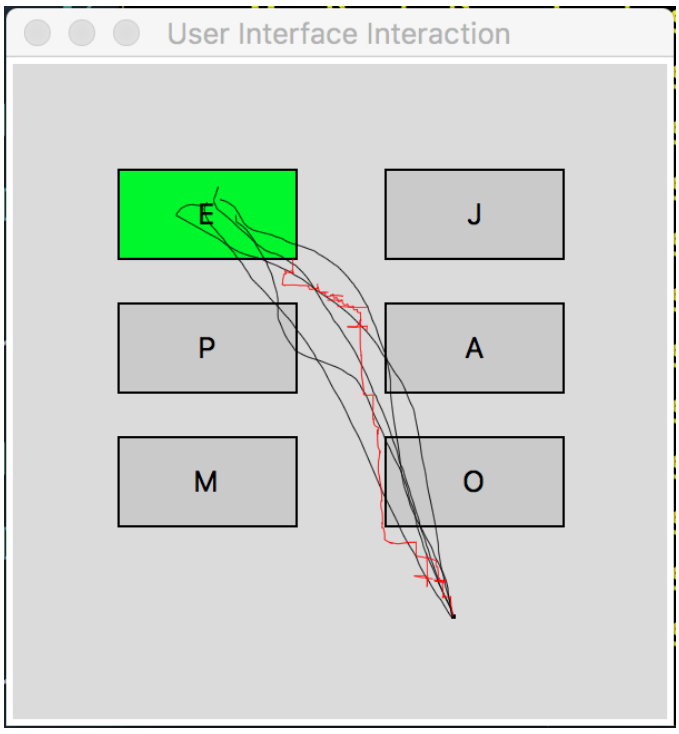


Figure 6-19: Test Case 10

6.3 Findings and Discussions

6.3.1 Fitts' Law Testing

With user interface interaction simulator, Fitts' Law can be compared or tested on it. Two tasks are simulated.

One Button User Interface – constant button size, various distances

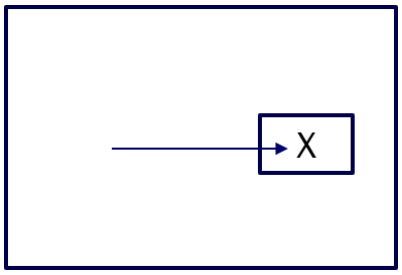


Figure 6-20: Task 1

Task 1 is performed with constant button size and various navigation distances.

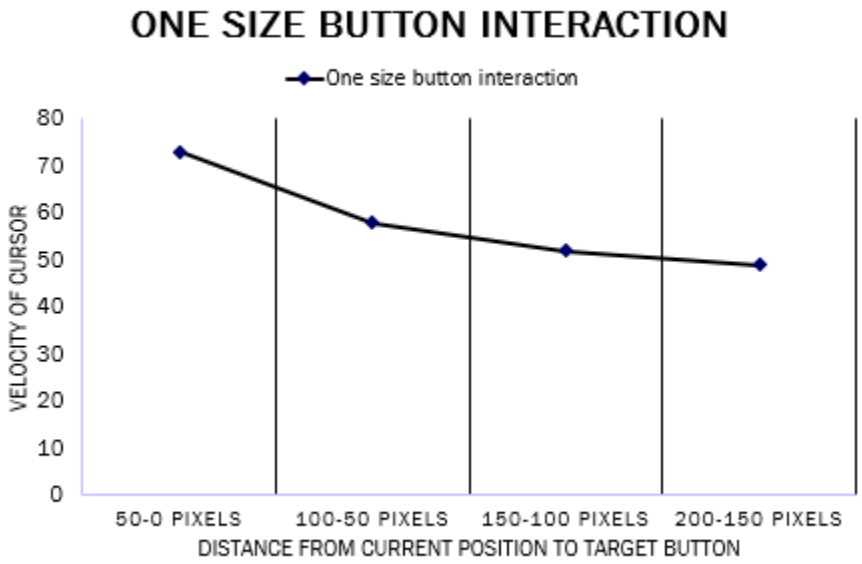


Figure 6-21: Average cursor velocity vs Distance from current position to target button

From Figure 6-21, it shows that average cursor velocity and distance from current position to target button is of linear relationship. Then

$$\frac{dS}{dt} = -C \cdot S \Longrightarrow -\frac{1}{\mathrm{C} \cdot \mathrm{S}}\mathrm{dS} = \mathrm{dt} \Longrightarrow \mathrm{t} = \frac{1}{\mathrm{C}}(\ln \mathrm{S} - ln\, S_0) \tag{6-1}$$

It indicates that the relationship between distance and navigation time satisfies Fitts' Law.

One Button User Interface – constant distance, various button sizes

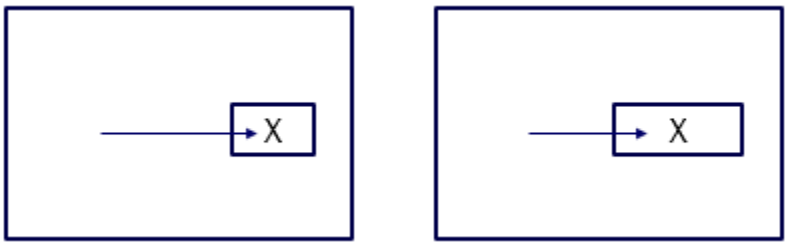


Figure 6-22: Task 2

Task 2 is performed with constant distance and various button sizes. Simulated interaction does not show correlations between the width of button and interaction steps. This test reflects one limitation of Phase II simulator that interaction simulation cannot simulate acceleration of cursor movement.

6.3.2 Avoidance of Non-target Buttons

In this verification study, each task is trained 2000 episodes due to time constraint. Agent might still need more training for more efficient interaction results. 10 test cases above all complete tasks in a neat way. In the verification results, Case 5 interacts with user interface outside of the human interaction result envelop. Following the interaction process with Case 5 agent, it can be found out that agent seems to avoid non targeted button to complete tasks. Similar behaviors can be also found in Case 8, 9 and 10. This phenomenon is strange in two ways:

1. No high rewards are founded near the boundary of the non-target widget.
2. Non-target widget avoidance is not expected to receive best accumulative rewards

This phenomenon can be explained with two possible reasons:

1. Not enough training is provided. From the Case 5 and cases with similar phenomenon, agent avoids passing non-target buttons. Passing other buttons achieves a more efficient way to gain more rewards but agent does not select it. It can be guessed that agent does not estimate true state value of position on non-target buttons. In previous interaction experience, agent is being punished by choosing Click option on non-target buttons which also influences agent's estimate to state value of other actions. Besides, policy of passing other button without any Click action is an event of low probability. In other words, agent is not very possible to try this policy and find it.

2. Neural network is a function approximation method with good feature of generality. Every time agent interacts with user interface by clicking non-target button, negative rewards might influence state value estimation around non-target button area.
3. In Phase II evaluation model, combination of collected interaction results is used to generate reward distribution for user interface interaction. If the reward density around the boundary of the non-target button is high, then it is possible to lead agent around the boundaries.

6.4 Quantitative Index for Analyzing User Interface: Suggestions and Future Research

In implementation methods, I have mentioned three quantitative indexes that can be used directly to reflect the interaction performance of a user interface and two manipulation methods. Three evaluation indexes are learning episodes, total interaction steps and proportion of stays to evaluate usability of a user interface. These quantities reflect one aspect of usability related to user interface interaction.

Learning episodes

Learning episodes refer to required number of episodes for agent to learn how to interact with user interface. Each episode agent keeps interacting with user interface until the completion of task. Each episode is basically one interaction strategy to complete a task. In exhaustive method, the number of total states can reflect the interaction complexity of a user interface. Reinforcement learning method accelerates learning process in comparison with exhaustive method but does not change the influence of interface complexity to learning process. Therefore, in evaluation model, quantity of learning episode is involved to reflect the complexity of a user interface.

Interaction steps

Interaction steps refer to total number of actions in one episode after training process is done or loss function converges. Inclusion of the action Stay and human interaction results as input data makes interaction steps be able to reflect relative length of time interacting with a user interface. Based on Phase II evaluation model data collection mechanism, speed of cursor is reverse proportional to the density of interaction. In area of cursor slowly moving, it has the high reward. In high reward area except target button, agent may choose to stay to gain more rewards which in turn reflects interaction time.

Proportion of action Stay

Following discussion of interaction steps, action Stay is most probably selected around high reward area. High reward area means slow cursor movement around. Therefore, proportion of stay can reflect levels of hesitation during interaction.

User interface suggestion

With these evaluation quantities, another question is whether it is possible to use these three quantities to suggest better performance user interface. Currently, the performance of user interface suggestion is quite limited. With simulated user interface, three evaluation quantities can be obtained. A common way to suggest improved user interface is to set different variables, simulate interaction results changing the value of these variables and make suggestions based on the change of the three quantities. There is a challenge in each of these steps.

1. Set different variables: A user interface consists of many different widgets. Each widget has many different feature to be considered. Widgets and their features are relatively independent. The input space of a user interface is huge and complicated.

2. Simulate interaction results changing the value of these variables: Large amount of interaction simulation consumes computational resources and requires long time to execute.
3. Make suggestions based on the change of the three quantities: This steps relates how to manipulate user interface and suggest a user interface with better usability. Simulated user interaction results can provide information related to interaction only. It does not consider anything about user interface design.

Third problem is big a barrier. I tested some cases of user interface suggestions with quantities and manipulation methods mentioned above. Two manipulation methods are considered as relatively ‘safe’ to use in avoiding unpredictable conditions in comparison of other free manipulation methods. However, during the test, it always happens that suggested user interface has overlapped widgets.

Suggested user interface with overlapped widgets cannot be used since human users do not know how to interact with overlapped buttons. Although it does not have practical use, it pushes the limit of interaction efficiency and makes a simple state that the best way to reduce the complexity level of user interface and shorten task completion time is to lay widgets of user interface together. The problem is human users cannot interact widgets laid up together only if the widgets on the surface are always the expected widgets.

In the future research, there can be two directions. One is to design a series of systematic manipulations to make changes to user interface. These manipulations are strictly restricted by well-defined rules fully considering human factors. Then simulated interaction results can be used to suggest improved user interfaces. Establishment of manipulation rules can be hard since factors that influence usability of a user interface are highly correlated.

Chapter 7 Summary

Chapter Summary

This chapter first gives a summary of the research. Then it talks about benefits and limitations of the research. At the end of this chapter, it discusses future research directions.

7.1 Summary of the Research

7.1.1 Phase I Summary

Phase I of the research is about static feature evaluation focusing on features such as the layout and dimensions of the widgets. It implements support vector machine to train classifiers for four different cases.

On the aspect of data collection, it explains the reasons for data asymmetry and provides a solution for future user interface subjective rating with combination of support vector machine and reinforcement learning method.

On the aspect of implementation method, it discuses the advantages and disadvantage of two ways to select separating hyperplane's insertion place. More important, it clarifies the relationship between kernel function and satisfaction and emphasizes the important connection between features of user interface and kernel function. It also indicates that to have good classification performance, it is necessary to find out kernel function the features of interest correspond to.

Phase I modeling results also suggest that the number of categories used for usability subjective ratings does not exceed three. Otherwise, some categories might be trivial or can be combined with adjacent categories.

Lastly, Phase I Modeling also provides some implications for user interface design:

1. Avoid obvious usability issues. As long as user interface does not exist obvious usability issues, normal users cannot sense the difference of small changes.
2. Widgets closer to the upper left corner have higher satisfaction levels.
3. Features satisfaction sensitivities ranked from from high to low are vertical dimensions, horizontal dimensions, horizontal positions and vertical positions. In the design of user interface widgets, extra cares are needed for those feature with high sensitivity.

7.1.2 Phase II Summary

Phase II of the research is mainly about dynamic interaction evaluation, focusing on dynamic features of user interface such as task completion time and task completion smoothness. It implements deep Q network to simulate human interaction behavior and uses interaction steps and proportion of action Stay as the evaluation quantities.

The interaction simulator is verified with envelop of human interaction data. A phenomenon is observed in the simulated results that agent shows the behavior of avoiding non-target widgets. This phenomenon is worth mentioning because based on the distribution of rewards agent is not expected to avoid non-target widgets. However, since the agent movement is a stochastic process, agent has the possibility to choose this option. This phenomenon is reasonable when viewed from the human side, since avoidance of crossing non-target buttons can reduce the error rate. This shows that the trained agent is generative in interacting with user

interface. Even if interaction does not show up in training data, agent can still explore and generate reasonable interaction results to mimic human's behavior.

Phase II modeling is then verified with Fitts' Law on the variable of target distance. As a major component of Phase II evaluation model, interaction engine simulates a simple task of cursor navigation and generated results satisfying Fitts' Law. This verification validates that Phase II evaluation results at least has value in simulating interaction task without complicated cognitive processing.

In Phase II modeling the reward distribution settings cannot simulate interaction movements in the second derivative order. Based on the research finding, the agent of Phase II modeling cannot well simulate cursor's acceleration. Selected evaluation quantities do not relate to acceleration of cursor. However, it is not flawless when the interaction engine simulating cursor movement within a small range.

7.2 Benefits and Limitations

7.2.1 Benefits

Phase I evaluation model has several benefits in comparison with traditional methods. First, Phase I modeling can reduce individual difference effects on evaluation. Users have their own preferences for certain interface features; however, these individual preferences cannot serve as general design guidelines for user interface. User interface evaluation is expected to find out common features shared by good user interfaces. In traditional heuristic evaluation method, individual preference and common features cannot be distinguished after evaluation, which reduces the reliability of the evaluation. A machine learning method is, in other words, a method

of pattern detection from big data analysis, where individual's preference or difference might be canceled. Second, Phase I modeling can increase the evaluation efficiency in comparison with traditional methods. Heuristic evaluation is to look at a user interface and brainstorm good or bad aspects of a user interface. This procedure is often time consuming and results are hard to interpret for design improvement. Phase I model, on the contrary, can evaluate a user interface in a short time without labor involved after the training process and also saves time.

Phase II evaluation model has some benefits in comparison with other evaluation models. First, Phase II evaluation model provides a systematic framework for user interface interaction organizing a stochastics model of interaction distribution and reinforcement learning methods. It provides a stochastic modeling for user interface interaction and combines the distribution of user interface interaction results with reinforcement learning rewards. Second, Phase II evaluation model can directly generated useful interaction results of user interface to describe the interaction. And it only needs human input as training data one time and does not need further human input. Third, Phase II evaluation model is an accumulative model. With more collected data, it can provide better prediction of user interface interaction results.

7.2.2 Limitations

Phase I evaluation method has two major limitations. First, Phase I model classifies user interface with support vector machine method that is difficult to use to revise evaluation standard for special user interface instances. Its implementation method decreases the flexibility of evaluation. Since all the evaluation factors or criteria are discovered through big data analysis, some factors may remain implicit that makes the evaluation standard adjustment more difficult. Second, divergence of the results may occur. If no proper kernel function is implemented, support vector machine may sometimes fail to work.

Phase II evaluation model also has some limitations. First, Phase II evaluation model does not take high dimensional sensory input. In Phase II evaluation model, states of cursors, position and dimension of widgets are all manually defined, which requires users to have some knowledge of machine learning. Second, Phase II evaluation model is not productive. Phase II evaluation model can only be trained to interact with one user interface and one task each time. User interface change and interaction change both lead to the need to repeat the learning process. Training time is around 15 to 20 hours for each individual task. Third, Phase II evaluation model cannot simulate interactions involving complicated cognitive activities such as memory, decision making and human errors. Phase II evaluation model does not involve any cognitive modeling. It is a computational model to analyze human-interface interaction behavior. In dealing with tasks involving cognitive activities beyond simple interactions, Phase II evaluation model is difficult to make useful predictions.

7.3 Future Research

Based on the discussion above, Phase I of the research leaves an important question for future research, that is, whether a scalar is enough to well describe the satisfaction level of usability. Modeling results have shown that for simple user interfaces a scalar rating can provide satisfying result to sufficiently describe satisfaction of a user interface. However, with the complexity of a user interface increases, this type of scalar rating seems weak. An assumption might be made that more complex user interfaces require more complex mathematical forms to describe them, such as a multi-dimensional vector or a function. Then more concrete questions

such as how to decide proper the mathematical form to describe satisfaction or usability and whether these forms can be integrated and unified for general user interface are all possible future research directions. Besides, during the data analysis we found that the number of satisfying user interfaces increases more slowly than linearity, which means that with the complexity of user interface increases the chance of getting high satisfaction user interface is getting smaller. This causes the asymmetric pattern of collected data and therefore leads to difficulty in data processing and analysis. It is also a challenge to adapt methods to asymmetric data.

Phase II of the research as discussed above has already built an interaction agent that has some generality. In future research, it could be possible to further develop interaction agent to fully equip it with learning ability. Also, as discussed in Chapter 6, task completion efficiency is constrained with the static view of user interface. For example, the most efficient user interface is to lay out widgets above each other, since it saves navigation time. However, this design cannot present necessary information to users to operate, and therefore it is not allowed by static evaluation. This shows there exists a tradeoff in user design. On one hand, user interfaces need to guarantee that users can clearly and efficiently perceive and understand information on the user interfaces. On the other hand, user interfaces need to guarantee users know how to output operations on user interface. One side is task completion efficiency and the other side is satisfaction of static view. How to balance these factors and maximize the total usability could be a good future study after this research.

Appendix

Supplementary Information

Support Vector Machine

High dimensional feature representation, non-linear relationship between inputs and outputs, extensive computational requirements, Phase I user interface evaluation is a challenging application of support vector machine. Among all those challenges, biggest challenge is the choice of kernel, the selection of the kernel function parameters. Good choice of kernel function is half way of succeeding in classifying. To help readers understand how kernel function works and its importance, in the following part of this section basic concepts of support vector machine will be introduced.

Suppose there are input and output sets, $\mathcal{X}$ and $\mathcal{Y}$ and training data set $(x_1, y_1), \cdots, (x_m, y_m)$. The purpose of support vector machine is to predict proper $y \in \mathcal{Y}$ given $x \in \mathcal{X}$, in other word, to find a function $f(\cdot)$ that $y = f(x, \alpha)$. α are parameters of the function. Through minimizing the overall risk:

$$\mathrm{R}_{\mathrm{emp}}(\alpha) = \frac{1}{m}\sum_{i=1}^{m} l(f(x_i, \alpha), y_i)$$

Linear Support Vector Machine:

For linearly separable datasets, support vector machine simply chooses a sets of hyperplane:

$$\frac{1}{m}\sum_{i=1}^{m} l(\omega \cdot x_i + b, y_i) + \left|\left|\omega\right|\right|^2$$

Subject to $min_i|\omega \cdot x_i| = 1$.

If dataset is separated perfectly, then problem becomes:

$$Minimize: \left|\left|\omega\right|\right|^2$$

Subject to:

$$(\omega \cdot x_i + b) \geq 1, \text{if } y_i = 1$$

$$(\omega \cdot x_i + b) \leq 1, \text{if } y_i = -1$$

These two constrains can be compacted to:

$$y_i(\omega \cdot x_i + b) \geq 1$$

Then classification function $f(\cdot)$ becomes a quadratic program problem.

If dataset is not separated perfectly, then problem becomes:

$$Minimize: \left|\left|\omega\right|\right|^2 + C\sum_{i=1}^{m} \xi_i$$

Subject to:

$$y_i(\omega \cdot x_i + b) \geq 1 - \xi_i, \xi_i \geq 0$$

A brief summary about formulation mentioned above:

Decision function:

$$f(\boldsymbol{x}) = \boldsymbol{\omega} \cdot \boldsymbol{x} + b$$

Primal formulation:

$$\text{min: } P(\omega, b) = \frac{1}{2}\left|\left|\omega\right|\right|^2 + C\sum_{i} H_1[y_i f(x_i)]$$

Hinge loss function $H_1(z) = \max\ (0, 1 - z)$

Non-linear Support Vector Machine:

Non-linear support vector machine always implements transformation Φ:

$$x \Longrightarrow \Phi(x)$$

Then find classification function $f(\cdot)$ that:

$$f(x) = \omega \cdot \Phi(x) + b$$

The decision function becomes:

$$f(x) = \sum_{i=1}^{m} \alpha_i \Phi(x_i) \cdot \Phi(x) + b$$

$\Phi(x) \cdot \Phi(x_i) = K(x_i, x)$ is called the kernel function.

Decision function is:

$$f(x) = \sum_{i=1}^{m} \alpha_i \Phi(x_i) \cdot \Phi(x) + b \; = \; \sum_{i=1}^{m} \alpha_i K(x_i, x) + b$$

Dual formulation is:

$$\text{min: } P(\omega, b) = \frac{1}{2} \left|\left| \sum_{\mathrm{i}}^{\mathrm{m}} \alpha_{\mathrm{i}} \, \Phi(\mathrm{x_i}) \right|\right|^2 + C \sum_i H_1[y_i f(x_i)]$$

Kernel function $K(\cdot,\cdot)$ is used to map from non-linear feature to linear feature. This is crucially important in building support machine. However, there is no clear rule about how to select kernel function. Some widely used kernel functions are:

1. Polynomial kernel: $K(x, x') = (x \cdot x' + 1)^d$
2. RBF kernel: $K(x, x') = e^{-\gamma \left|\left| x - x' \right|\right|^2}$

Deep Q Network

Deep Q network (always known as DQN) is a variants of reinforcement learning. Different with traditional reinforcement learning, DQN substitute Q table with a neural network in order to solve infinite possible states. In the following of this section, I will introduce basic structure of deep Q network.

Reinforcement learning provides a framework for decision making and deep learning provide a frame work for pattern learning. DQN is a combination of decision making and pattern learning, which generate to some degree a better AI.

Before understanding how reinforcement learning works, three concepts need to be introduced first, policy, value function and model.

Policy represents for agent's behavior function mapping from state to action.

Deterministic policy:

$$a = \pi(s)$$

Stochastic policy:

$$\pi(a|s) = P[a|s]$$

Value function is a prediction or estimation for future reward.

From state s and action a under policy π, value function can be represented as:

$$Q^{\pi}(a|s) = E[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots |s, a]$$

Value function decompose into a Bellman equation:

$$Q^{\pi}(a|s) = E_{s',a'}[r + \gamma Q^{\pi}(a'|s')|s, a]$$

To maximize the total expected future rewards:

$$Q^{*}(s, a) = max_{\pi} Q^{\pi}(s, a) = Q^{\pi^{*}}(s, a)$$

Then

$$\pi^*(s) = argmax_a Q^*(s,a)$$

Formally, optimal values decompose into a Bellman equation,

$$Q^*(s,a) = E_{s'}[r + \gamma max_{a'} Q^*(s',a')|s,a]$$

Model is agent's representation of the environment and is learnt from experience.

To solve traditional Q learning problem, minimize loss function:

$$l = \left(r + \gamma \max_a Q(s',a',\omega) - Q(s,a,\omega)\right)^2$$

If the result converges, value table for under actions and states can be obtained. Also the optimization may be divergent because of correlations between samples or Non-stationary targets.

To remove correlations, build data-set from agent's own experience.

$$l = \left(r + \gamma\, max_{a'} Q(s',a',\omega^-) - Q(s,a,\omega)\right)^2$$

Remove upward bias caused by $\max_a Q(s',a',\omega)$

$$|r + \gamma max_{a'} Q(s',a'.\omega\_) - Q(s,a,\backslash omega)|$$

Also, to split Q-network into two channels:

$$Q(s,a) = V(s,v) + A(s,a,\omega)$$

$$V(s,v) \approx E[r_{t+1} + r_{t+2} + \cdots |s]$$

$$Q(s_t,a_t) = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n},v)$$

Actor is updated towards target

$$\frac{\partial l_u}{\partial u} = \frac{\partial\, log\, \pi\,(a_t|s_t,u)}{\partial u}\left(Q(s_t,a_t) - V(s_t,v)\right)$$

Critic is updated to minimize MSE

$$l_v = \left(Q(s_t,a_t) - V(s_t,v)\right)^2$$

Bibliography

Becker, S., & Hinton, G. (1992). A Self-organizing neural network that discovers surfaces in random-dot stereograms. 355. Nature.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Tr. Neural Nets.

Bennett, J. L. (1979). The commercial impact of usability in interactive systems. In B. Shackel, *Man Computer Communication, Infotech State of the Art, vol. 2* (pp. 1–17). Maidenhead: Infotech International.

Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*.

Bias, R. G. (1994). The pluralistic usability walkthrough: coordinated empathies. In J. Nielsen, & R. L. Mack, *Usability inspection methods* (pp. 63 - 76). John Wiley & Sons, Inc.

Card, S., Newell, A., & Moran , T. (1983). *The Psychology of Human-Computer Interaction.* Hillsdale: L. Erlbaum Associates Inc.

Dean, T., & Jabazawa, K. (1989). A model of reasoning about persistence and causation. *Computational Intelligence*.

Desurvire, H. W. (1994). Faster, cheaper!! Are usability inspection methods as effective as empirical testing? In H. W. Desurvire, *Usability inspection methods* (pp. 173 - 202). John Wiley & Sons, Inc.

Desurvire, H., Kondziela, J., & Atwood, M. E. (1992). What is gained and lost when using methods other than empirical testing. *CHI '92 Posters and Short Talks of the 1992*

SIGCHI Conference on Human Factors in Computing Systems (pp. 125-126). Monterey: ACM.

Elman, J. (1993). Learning and development in neural networks: the importance of starting small. *Cognition*.

Ericsson, K., & Simon, H. (1980). Verbal Reports as Data. *Psychological Review*, 215-251.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of patern recognition unaffected by shift in position. *Biological Cubernetics*.

Giudice, M., Manera, V., & Keysers, C. (2009). Programmed to learn? The ontogeny of mirror Neurons. . *Dev. Sci.*, .

Gray, W., & Salzman, M. (1998). Damaged merchandise? A review of experiments that compare usability evaluation methods. *Human-Computer Interaction*, 203-261.

Hastad, J., & Goldmann, M. (1991). On the power of small-depth threshold circuits. *Computational Complexity*.

Hinton, G. (1989). Connectionist learning procedures. *Artificial Intelligence, .*

Jacobsen, N. E., & John, B. (1998). THE EVALUATOR EFFECT IN USABILITY STUDIES: PROBLEM DETECTION AND SEVERITY JUDGMENTS. *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting* (pp. 1336-1340). Chicago: HFES.

Jeffries, R., & Desurvire, H. (1992). Usability testing vs. heuristic evaluation: was there a contest? *ACM SIGCHI Bulletin Volume 24 Issue 4* (pp. 39 - 41). ACM.

Jeffries, R., Miller, J. R., Wharton, C., & Uyeda, K. (1991). User interface evaluation in the real world: a comparison of four techniques. *CHI '91 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 119-124). Louisiana: ACM.

Jordan, M. I. (1998). *Learning in Graphical Models.* Dordrecht: Kluwer.

Kahn, M. J., & Prail, A. (1994). Formal usability inspections. In J. Nielsen, & R. L. Mack, *Usability inspection methods* (pp. 141 - 171). John Wiley & Sons, Inc.

Kirkpatrick, S., Jr., C., & Vecchi, M. (1983). Optimization by simulated annealing. *Science*.

L' Ecuyer, P. (1994). Efficiency improvement and variance reduction. *Proceedings of the 1994 Winter Simulation Conference.* Montreal: University of Montreal.

Lewis, J. (1994). Sample Sizes for Usability Studies: Additional Considerations. *Human Factors*, 368-378.

Memisevic, R., & Hinton, G. (2007). Unsupervised learning of image transformations. *Proceedings of the Computer Vision and Pattern Recognition Conference.* Toronto.

Miller, R. B. (1971). *Human ease of use criteria and their tradeoffs.* Poughkeepsie: IBM, Systems Development Division, Poughkeepsie Lab.

Mitchell, T. (1997). *Machine Learning.* New York: McGraw-Hill.

Moller, M. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*.

Nielsen, J. (1994). Enhancing the explanatory power of usability heuristics. *CHI '94 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 152-158). ACM.

Nielsen, J. (1994). Usability Inspection Methods. *CHI '94 Conference Companion on Human Factors in Computing Systems* (pp. 413-414). Boston: ACM.

Nielsen, J., & Molich, R. (1990). Heuristic Evaluation of User Interfaces. *ACM CHI 90 Human Factors in Computing Systems Conference* (pp. 249-256). Washington: ACM.

Novick, D. G., & Hollingsed, T. (2007). Usability Inspection Methods after 15 Years of Research and Practice. *SIGDOC '07 Proceedings of the 25th annual ACM international conference on Design of communication* (pp. 249-255). El Paso: ACM.

Olshausen, B., & Field, D. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*.

Russel, S. J., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach.* Upper Saddle River: Prentice Hall,.

Scholtz, J. (2004). Usability Evaluation. *National Institute of Standards and Technology*, Volume 1. Retrieved from http://notification.etisalat.com.eg/etisalat/templates/582/Usability%2520Evaluation_rev1[1].pdf

Shackel, B. (1959). Ergonomics for a Computer. *Design I20*, 36 - 39.

Shackel, B. (1990). Human factors and usability. In J. Preece, *Human-computer interaction* (pp. 27-41). Upper Saddle River: Prentice Hall.

Sietsma, J., & Dow, R. (1991). Creating artificial neural networks that generalize. *Neural Networks*.

Smith, S. L., & Mosier, J. N. (1986). *GUIDELINES FOR DESIGNING USER INTERFACE SOFTWARE.* The MITRE Corporation.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning - An Introduction.* Cambridge: MIT Press.

Taylor, G., Hinton, G., & Roweis, S. (2006). Modeling human motion using binary latent variables. *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems.* Vancouver: Advances in Neural Information Processing Systems 19.

Tullis, T., Fleischman, S., Mcnulty, M., Cianchette , C., & Bergel, M. (2002). An empirical comparison of lab and remote usability testing of websites. *Proceedings of Usability Professionals Conference.* Boston.

Von Melchner, L., Pallas, S., & Sur, M. (2000). Visual behaviour mediated by retinal projections directed to the auditory pathway. *Nature*.

Wharton, C., Rieman, J., Lewis, C., & Polson, P. (1994). The cognitive walkthrough method: a practitioner's guide. In J. Nielsen, & R. L. Mack, *Usability inspection methods* (pp. 105 - 140). John Wiley & Sons, Inc.

Williams, R., & Zipser, D. (1989). A learning algorithm for continually running fully recurrrent neural networks. *Neural Computation*.

Yu, D., Wang, S., & Deng, L. (2010). Sequential labeling using deep-structured conditional random fields. *IEEE Journal of Selected Topics in Signal Processing*.