User Experience Evaluation through Automatic A/B Testing

Juan Cruz Gardey
LIFIA, Fac. de Informática, Univ. Nac. de La Plata
La Plata, Argentina
jcgardey@lifia.info.unlp.edu.ar

Alejandra Garrido
LIFIA, Fac. de Informática, Univ. Nac. de La Plata &
CONICET
La Plata, Argentina
garrido@lifia.info.unlp.edu.ar

ABSTRACT

The goal of this research is to develop an A/B testing method to automatically compare the user experience (UX) of alternative designs for a web application in a real context with a large number of users. The challenge that it poses is to find mechanisms to predict the UX with machine learning techniques. This submission outlines the motivation, research goal, current status and remaining work.

CCS CONCEPTS

• **Human-centered computing** → **HCI design and evaluation methods**; • **Computing methodologies** → Machine learning approaches.

KEYWORDS

User Experience; User Interaction; A/B testing; Machine Learning

ACM Reference Format:
Juan Cruz Gardey and Alejandra Garrido. 2020. User Experience Evaluation through Automatic A/B Testing. In *25th International Conference on Intelligent User Interfaces Companion (IUI '20 Companion), March 17–20, 2020, Cagliari, Italy.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3379336.3381514

1 INTRODUCTION

While the UX of a web application is a crucial factor that determines its success, UX evaluation is often neglected, mainly due to its high cost. Although there are several proposals to reduce the costs of UX evaluation such as remote user testing [1] and automatic user interaction analysis [7], as Garcia et. al.[2] evidences, there is a lack of tools and methods to mediate the collaboration between designers and developers in the UX evaluation of the application being developed, which implies not only discovering the UX issues, but also the identification and the evaluation of alternative designs.

This research is focused on providing support for evaluating alternative designs of the user interface (UI) of a web application. Specifically, we are investigating a method similar to A/B testing to allow designers to compare the UX of different designs and determine the best solution for a given set of UX problems in the target application. A/B testing (also called split testing) is a well-known technique that consists of deploying two or more versions of a UI [8]. Users are split between the versions in a persistent manner, and key metrics are computed to determine the best version in terms of "conversion rate", which refers mostly to increased revenue.

Even though most of the commercial A/B testing tools such as Google Optimize [4] and Optimizely [9] are intended to evaluate conversion rate, there are research works that propose a variation of A/B testing to evaluate usability [10] or user's preference [11]. Both approaches require user input to compare the alternative designs, which can hinder the evaluation in a real application, lacking transparency for the subjects. The split testing proposed in [10] relies on user interaction analysis and user questionnaires to train models to predict the usability of a UI. These models are specific for the application under test. On the other side, the evaluation developed in [11] is not conducted on a working application, but on screenshots or mockups that are randomly presented to the users for them to select the preferred one. Another aspect lacking attention in the literature is the technology needed to set up the alternative designs to be tested, which is very relevant to prevent the cost of a manual implementation.

Contrarily to previous approaches, we are interested in automatically evaluating alternative designs in terms of UX, i.e., without requiring explicit users intervention, in order to easily conduct the evaluation in a real context with a large number of users. In this sense, we propose the development of Machine Learning (ML) models to assess the UX of a design. In a recent work we approached this proposal by predicting the user interaction effort with some UI elements through the analysis of user interaction metrics automatically captured [6]. Using the user interaction effort as a measure of UX, the next step is to define a method to compare alternative solutions to UX problems. Besides that, to make the A/B testing affordable, support must be provided from the beginning of the evaluation, which starts by configuring the designs to be tested.

2 RESEARCH GOAL

The goal of this research is to develop a method to conduct A/B testing experiments to automatically compare the UX of alternative designs of a web application. The evaluation method is aimed to support designers in the task of UX improvement once a set of UX issues have been discovered, and should work directly on production in a transparent way for the users. This research addresses two different aspects:

- Set up the web application's versions to be tested. In order to allow designers to work on their own, we are investigating a method to create alternative versions of a web application without altering its source code, i.e, by directly modifying its UI in the browser.
- Compare different versions of a web application. To support designers in finding the version that provides a better UX,

IUI '20 Companion, March 17–20, 2020, Cagliari, Italy

ACM ISBN 978-1-4503-7513-9/20/03.
https://doi.org/10.1145/3379336.3381514

our goal is to develop a method and a tool to automatically evaluate the UX of these versions by analyzing user interaction events. To this end, we believe that ML models are an important technique to predict users' behaviour because its multiple interaction patterns involved cannot be easily analyzed defining heuristics.

3 CURRENT STATUS

We have been working on both aforementioned goals. Our proposal to create alternative versions of a web application is based on the notion of *Client-side web refactorings* (CSWRs) [3]. Each CSWR is a predefined transformation that performs small changes in the client of a web application through a script, like adding or replacing a widget, with the aim of improving the UX while preserving functionality. Our first approach consisted in a tool that allows creating a web application version by applying CSWRs semi-automatically, once usability problems have been found by analyzing user interactions [5]. In that case, the analysis of user interaction events was based on heuristics, and the tool requires collecting a large amount of interaction events to be able to discover usability problems and suggest CSWRs that may solve them. The evaluation of this tool suggested the need of a method intended for designers to freely explore potential solutions to UX issues through CSWRs, i.e, by directly selecting an element from the UI and a CSWR to apply on it. In this sense, we are working on a visual programming tool called UX-Painter, a web-extension that allows designers to work directly on production. Currently, we are conducting experiments with UX designers to assess its validity.

With respect to the evaluation of designs, since we propose the application of CSWR to create alternative UIs, our first step was to investigate a method to measure the effectiveness of the changes applied by the CSWRs in the context of a specific application. Thus, we developed a unified score called *interaction effort*, to assess the level of effort required by users to interact with different widgets [6]. Since alternatives CSWRs for a specific UX problem apply different changes, the proposed score aims to compare how different widgets perform for the same task. This score is predicted from micro-measures automatically captured from interaction logs, that are specific to each widget type. To be able to predict the interaction effort, we have to analyze users interacting in real web applications in two different moments: in a preliminary study to discover the micro-measures that influence the user's interaction effort for each widget type, and then with the micro-measures defined to obtain their samples to train each widget's prediction model. We already conducted the preliminary study to define the micro-measures that determine the interaction effort of two widget types, text inputs and selects, and we have validated this approach obtaining interaction samples and manually ranking them to train decision tree classifiers [6]. We decided to use decision trees as preliminary models to get feedback about the importance of each micro-measure in the resulting classification. The first results achieved allow us to go further in the interaction effort prediction, namely collect more user interaction samples and evaluate the performance of others prediction models.

4 NEXT STEPS

Concerning the setup of the designs to be tested, the next step is to develop a method to deploy in production the alternative versions created with UX-Painter and to split the users between them consistently. Besides that, we plan to incorporate new CSWRs to UX-Painter to offer designers different solutions for each UX issue considered.

Regarding the evaluation of alternative versions, we are creating tools to support the collection of training data, specifically, to capture the micro-measures of each widget interaction and to allow UX experts to easily rank them by looking at recorded screencasts. This will enable approaching our next challenge, i.e., to obtain a large amount of samples to implement the prediction models of all the widgets involved in the CSWRs. Also, this requires analyzing which ML model works better with the collected dataset. After being able to obtain the user's interaction effort of all widget types, we still need to define a strategy to determine which of the versions under test is better in terms of UX. The interaction effort score judges the interaction of a single user in a specific widget of a page, so first we need to define an overall interaction effort of a specific widget that considers all the user interactions gathered during the evaluation on that widget. Moreover, considering that each tested version may include different CSWRs, a method to weight the interaction effort of the different widgets is required to obtain a measure of the overall UX of each version to identify the best one. Last but not least, we may also consider other elements to judge the UX besides the interaction effort metric, by other ML models that may predict users' comfort, joy, or other aspects involved in the UX.

ACKNOWLEDGMENTS

The authors acknowledge the support from the Argentinian National Agency for Scientific and Technical Promotion (ANPCyT), grant number PICT-2015-3000.

REFERENCES

[1] Paolo Burzacca and Fabio Paternò. 2013. Remote usability evaluation of mobile web applications. 8004 LNCS, PART 1 (2013), 241–248.
[2] Andrei Garcia, Tiago Silva da Silva, and Milene Silveira. 2019. *Artifact-Facilitated Communication in Agile User-Centered Design*. Vol. 2. Springer International Publishing. 260 pages.
[3] Alejandra Garrido, Sergio Firmenich, Gustavo Rossi, Julian Grigera, Nuria Medina-Medina, and Ivana Harari. 2013. Personalized web accessibility using client-side refactoring. *IEEE Internet Computing* 17, 4 (2013), 58–66.
[4] Google. 2020. Optimize. https://www.optimize.google.com
[5] Julian Grigera, Juan Cruz Gardey, Alejandra Garrido, and Gustavo Rossi. 2018. Live versioning of web applications through refactoring. In *ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*.
[6] Julian Grigera, Juan Cruz Gardey, Andres Rodriguez, Alejandra Garrido, and Gustavo Rossi. 2019. One metric for all: Calculating interaction effort of individual widgets. In *Conference on Human Factors in Computing Systems - Proceedings*.
[7] Julián Grigera, Alejandra Garrido, José Matías Rivero, and Gustavo Rossi. 2017. Automatic detection of usability smells in web applications. *International Journal of Human Computer Studies* 97, September 2016 (2017), 129–148.
[8] Ron Kohavi and Roger Longbotham. 2015. Online Controlled Experiments and A/B Tests Motivation and Background. *Encyclopedia of Machine Learning and Data Mining* Ries 2011 (2015), 1–11.
[9] Optimizely. 2019. Optimizely. https://www.optimizely.com
[10] Maximilian Speicher, Andreas Both, and Martin Gaedke. 2014. Ensuring Web Interface Quality through Usability-Based Split Testing. *Icwe, LNCS 8541* (2014), 93–110.
[11] Jean Vanderdonckt, Mathieu Zen, and Radu Daniel Vatavu. 2019. AB4Web: An on-line A/B tester for comparing user interface design alternatives. *Proceedings of the ACM on Human-Computer Interaction* 3, EICS (2019).