A Long-Term Evaluation of Adaptive Interface Design for Mobile Transit Information

Oscar J. Romero
oscarr@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA

Alexander Haig
katadh@sbcglobal.net
Carnegie Mellon University
Pittsburgh, PA

Lynn Kirabo
lkirabo@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA

Qian Yang
yangqian@cmu.edu
Carnegie Mellon University
Pittsburgh, PA

John Zimmerman
johnz@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA

Anthony Tomasic
tomasic@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA

Aaron Steinfeld
steinfeld@cmu.edu
Carnegie Mellon University
Pittsburgh, PA

ABSTRACT

Personalization of user experience has a long history of success in the HCI community. More recently the community has focused on adaptive user interfaces, supported by machine learning, that reduce interaction efforts and improves user experience by collapsing transactions and pre-filtering results. However, generally, these more recent results have only been demonstrated in the laboratory environment. In this paper, we share the case of a deployed mobile transit app that adapts based on users' previous usage. We examine the impact of adaptation, both good and bad, and user abandonment rates. We conducted an 18-month assessment where 2,616 participants (with and without vision impairments) were recruited and participated in an A/B study. Finally, we draw some insights on some unusual effects that appear over the long term.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in interaction design**.

KEYWORDS

Adaptive interfaces; mobile systems; accessibility; transportation

ACM Reference Format:
Oscar J. Romero, Alexander Haig, Lynn Kirabo, Qian Yang, John Zimmerman, Anthony Tomasic, and Aaron Steinfeld. 2020. A Long-Term Evaluation of Adaptive Interface Design for Mobile Transit Information. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '20), October 5–8, 2020, Oldenburg, Germany*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3379503.3403536

MobileHCI '20, October 5–8, 2020, Oldenburg, Germany

ACM ISBN 978-1-4503-7516-0/20/10.
https://doi.org/10.1145/3379503.3403536

1 INTRODUCTION

Many popular mobile applications use personalization to improve the user experience. For example, shopping applications employ recommender systems to reduce the amount of searching the user must do. Likewise, most map apps allow users to create bookmarks for popular destinations, like home or work, to reduce future interaction effort. In this interaction design, the app will rearrange the order of possible selections based on past knowledge of or explicit setting of preferences by the user.

As seen in the commercial product space, the research community has also not deeply explored the value of improving user experience through adaptive interfaces. The HCI research community has performed a lot of research and technical development towards this area [19], so why are adaptable user interfaces so rare? One likely cause might be a lack of evidence that adaption really improves user experience. There is also mixed results in the literature on the value of adaptation. Some research shows it can have a negative impact (e.g. Microsoft Smart Menus [7]). Almost no studies show the impact on long-term use. Typically, a study simply measures things like short time savings, or lower GOMS scores (e.g., [27]). Another reason may be that the machine learning –ML– models that support adaption perform well in the lab, but the models do not perform well when transferred into the real world.

We wanted to understand if UI adaptation that reduces interaction effort can improve user experience in the long run. In particular, we were interested in how adaptation could support users with disabilities who normally have trouble managing large amounts of data on mobile interfaces (e.g., scrolling through a long list with a screenreader, etc.). To this end we designed Tiramisu, an adaptive mobile transit app that shows bus arrival information. It adapts the information display by predicting and filtering for the bus routes users are most likely interested in using time of day and location. We deployed this application to the public and ran an 18-month A/B study on users of the application (2,616 users) in a medium-sized city in North America. We gathered and analyzed user navigation

behavior within the app and bus route filter selection behavior. This paper makes four contributions to the literature:

(1) We provide empirical evidence from a field deployment on the impact of adaptation in a mobile application that reduces user selection and navigation effort. We analyze this evidence to understand the impact of adaptation on interaction effort and abandonment rate of the application.
(2) We provide an example of an adaptive mobile interface design that collects ground truth labels for training ML and logging transit behaviors. We also discuss the challenges encountered with this design approach.
(3) We provide an example of a (very) long-term assessment of an application deployed in the real world, and some unusual effects that appear over the long term.
(4) We provide additional support for using deployed apps as effective research testbeds, especially for hard to recruit user populations like people who are blind or low vision.

This paper is organized as follows: Section 2 details the interaction design, Section 3 describes the A/B study, Section 4 reports the findings, Section 5 presents the discussion, Section 6 describes the related work and Section 7 presents the conclusions.

2 INTERACTION DESIGN

Adaptive UI design lends itself to situations where users (i) repeatedly perform the same or similar interactions over time, (ii) the interaction is burdensome, and (iii) sufficient evidence is available such that the interaction can be learned and accurately predicted (usually by a ML algorithm). We chose an application where the three conditions apply.

2.1 Context

Users of a mobile transit information app regularly interact with the application with the same information seeking task, namely to discover the best (next) bus for the current trip. This application seems like a perfect place for an adaptive interface for several reasons: (i) many people likely want the same information (based on time-day-location) when using transit because they travel in repeated patterns while commuting; (ii) without filtering, the UI can easily have over 100 rows of transit choices in response to a particular request (particularly in dense metro areas); (iii) for some urban trips, there are generally multiple possible bus routes to take; and (iv) the user's location, date, and time are readily available. With this evidence, each user can have an individualized ML model.

2.2 Design

Our design goal was to make an adaptive user interface in a mobile transit information application that reduced the amount of bus route filter selection, scrolling, reading of screens, and other navigation. In turn, this reduction makes the application easier to use and thus the application would theoretically have a lower rate of abandonment (compared to the non-adaptive case). We started the design process by identifying opportunities and patterns for UI adaptation. We determined that the following were possible: multiple landing pages, learning multi-screen transactions, and learning the filters that users applied to scrolling. We settled on multiple landing pages and automatic pre-filtering as adaptation features (see Figure 1).

For the landing page, the app was redesigned to skip the normal landing page (which lists several options for the user) and automatically select the transit information option, as proposed by [26]. Then, the system automatically computed the most likely bus stop(s) of the user through the use of location information, eliminating the need to draw a map of the local area and the need for the user to select a stop on the map. For this adaption, no ML is used, since a heuristic based on the closest stop locations to the user's current location was sufficiently accurate to use all the time.

The system automatically pre-filtered transit information shown to the user on the main listing of buses arriving at a location. This pre-filtering, if done correctly, would reduce the interaction effort required to obtain information. To accomplish this, the system had to predict both the (i) routes and (ii) IN-to-town/OUT-of-town direction desired by the user. To accurately predict routes and direction, the system needed context-awareness: the use of the context-sensitive information of user identity, location, date, and time. Given context awareness and a history of the prior filtering choices of the users, a ML model was constructed to predict desired routes and direction. This model was then executed every time the user visited the transit information page and its results were used to pre-filter routes and direction. Based on the pre-filter, the initial listing of buses was displayed to the user. At the top of the screen, the system provided a list the filters selected on and off, allowing the user to quickly correct any mis-predictions by the system (and obviously providing additional data for the model to learn).

An initial study of ML performance indicated that the best performance was achieved with a personalized model. This fact was not surprising since any two individuals at a bus stop on a particular date and time do not have the same information need. However, personalized models creates a cold-start problem. When a new user joins the application, no data is available to build a model for the user. Thus, we adapted a two-phase strategy that triggered adaptation based on the amount of available information. In the first phase (when no historical data is available) pre-filtering relied mostly on current time and user's location (context-awareness), thus mitigating the cold-start problem. In the second phase, pre-filtering relied mainly on a model that learned the user's transit behaviors to predict what bus information the user wanted.

2.3 Machine Learning Model

ML was utilized to automatically filter out information on arriving transit vehicles based on information about the user's current context. To this end, we used a Random Forest model [1], a simple but powerful supervised ML method that combines many weak learners (decision trees) into strong learners improving prediction accuracy. If available, our Random Forest model employs the user's previous behavior to predict route filters. Otherwise, it makes filter predictions based on traffic behaviors observed in a group of similar users. We defined a 30-tree-estimator classifier per user which was trained on user logs, where selection/deselection of route filters were used as a means to label correct/incorrect adaptations. That is, every time the user selected/deselected a route filter (labels), we logged the time, date, day of the week, location, and any selected inbound/outbound bus filter (features). We then used these features and labels to train the ML model so it could predict which routes and direction the user would select. When users who received a

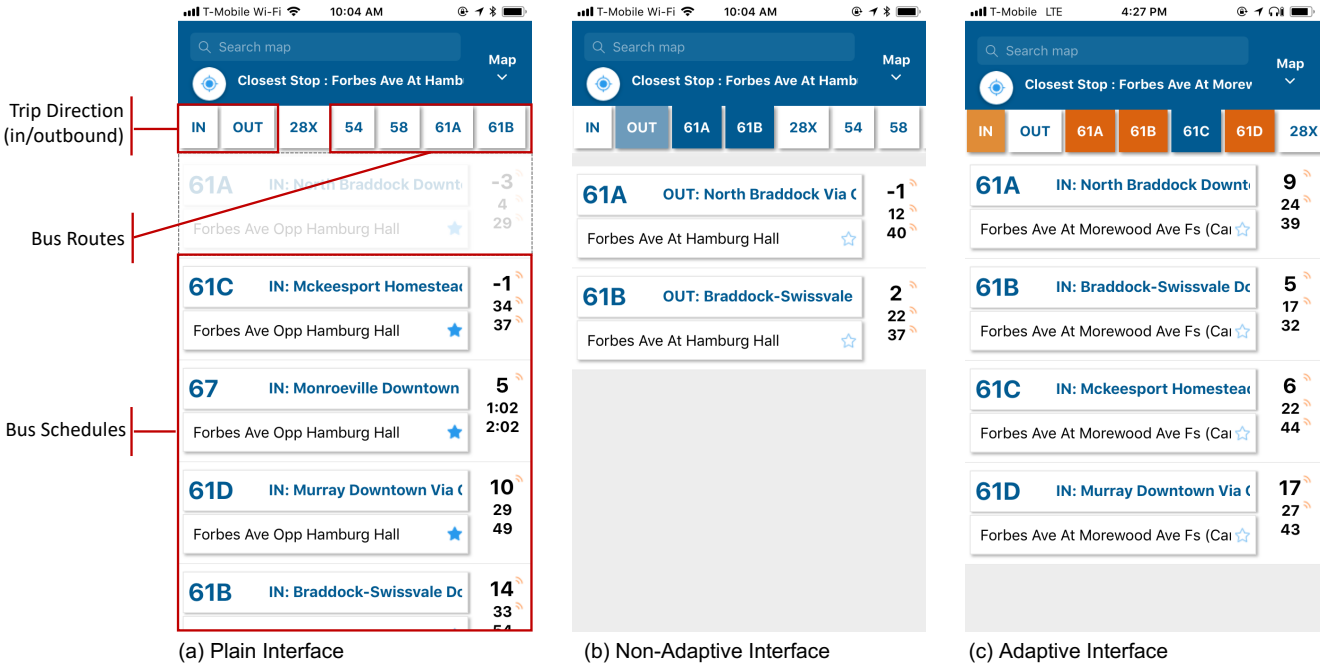


Figure 1: App User Interface. a) Plain Interface: neither the user has selected nor the system has predicted any bus route, so the user has to scroll down to see all the results. The location shown in this figure has 14 bus schedule rows for 8 routes. b) Non-Adaptive Interface: user selects bus routes 61A and 61B for the outbound direction (blue boxes) to filter out the results, but there is no adaptation. c) Adaptive Interface: Interface adaptation is implemented in the form of prioritized results in our app. The top section of the results window contains the most frequently routes the user takes given the current context (time and location) filtered by either inbound or outbound direction, i.e., bus routes 61A, 61B, and 61D. Filter buttons are colored differently. Orange shows the actions taken by the app, the predicted filters. Blue shows filters selected by the user.

prediction did not perform any filter selection/deselection action, then we considered the prediction to be correct. When users selected additional filters, then we captured this as new training data (retrained every 24 hours). When users deselected predicted filters, we considered this a mislabeling error. At least 80 samples (filter selections) were needed in order for the Random Forest algorithm to make a route prediction for a specific user.

We used the Random Forest algorithm because it reduces dramatically mislabeling errors and over-fitting by means of voting and averaging the results. Also, Random Forest is very well-known for outperforming other algorithms (ANN, SVM, Boosted Trees) in terms of accuracy, AUC-ROC curve, and increasing dimensionality [4]. Though the feature vector of our classifier is rather small at this moment, we anticipate that we will need to increase the number of dimensions once we incorporate additional features in the app and model. Thus, Random Forest is an excellent candidate to scale efficiently to those high dimensions.

2.4 Implementation

In order to reach a broader audience for Tiramisu app, we decided to make it available on both Android and iOS platforms[1]. To this end, we used *Ionic*, a development framework that allows for building cross-platform apps. For the back-end, we set up instances of Amazon Web Services (AWS) that performed diverse tasks: running the ML model, logging/storing all the user interaction, and sending requests to *OneBusAway*, a public API that gets real-time transit information on scheduled and predicted bus arrival times.

[1]http://www.tiramisutransit.com/

3 A/B STUDY DESIGN

We used an A/B study to assess the long-term impact of the adaptation and compared participants in two conditions.

3.1 Participants

We ran our study on participants from a medium-sized city in North America. Participants downloaded our app voluntarily and were randomly assigned to one of the two following conditions:

Adaptive Condition (AC): The UI in this condition adapted to participant needs by filtering the display of the bus routes and trip direction (inbound/outbound). ML training was a continuous process based on user filter selections over time).

Non-Adaptive Condition (NAC): Participants in this condition received the same app, but the system made no attempt to automatically filter bus routes or trip direction.

3.2 Data Collection

Our app was heavily instrumented to record user interactions. Logging included details about context-sensitive information like when and where the app was used, what information users requested

and selected, and which features they utilized within the app. We also logged whether the screen reader was turned on, which is a likely indicator that the user was blind or low vision. We logged user interactions during an 18-month window from 01 July 2018 to 31 December 2019. The dataset was pre-processed to remove incomplete, inaccurate, or corrupted items (described below).

3.3 Hypotheses

We propose several hypotheses drawn from both the literature aforementioned review and empirical data:

Hypotheses about Effort:

H1: Participants in AC will select fewer filters and make fewer navigation actions (scroll less) than participants in the NAC. This is because the adaptive interface should effectively predict and execute user selection actions, thus reducing the effort in terms of the number of selections a user needs to make.

H1.1: When taking into account the performance of the ML model, participants in the AC will select fewer filters and make fewer navigation actions, even when the model generates incorrect route predictions.

H1.2: Participants who use the screen reader (likely blind or low vision) in the AC will select fewer filters and make fewer navigation actions than sighted participants in the NAC. This is because the adaptive interface should reduce friction and effort.

Hypotheses about Abandonment:

H2: Participants in the NAC will abandon the app at a higher rate than participants in the AC because the adaptive interface will reduce the effort to use it over time making it be more pleasurable to use than a non-adaptive app.

To test H1, we filtered out participants who never used a single filter across all their sessions. To test H2, we discarded participants who left the app open for several weeks and did not interact with the app; we could not determine whether they abandoned the app.

3.4 Method

We utilized a randomized A/B testing protocol. This approach, widely used in industry, randomly assigns different user experiences to groups of users for comparative analyses. We used this method to compare pre-filtering (AC) with no filtering (NAC) conditions. We did not follow up our A/B with targeted interviews for several reasons: (i) the system was deployed in the real world, not in a lab, so we did not have fine-grained control of when and how participants used the app; (ii) since we were manipulating privacy-sensitive information (location history, transit behaviors, etc.), we decided to keep users anonymous and did not collect personal information; and (iii) gathering the user's common behaviors through a series of Internet forms was rejected as too burdensome.

3.5 Metrics

We were interested in measuring effort reduction and abandonment rate using a set of specific metrics. Further analysis was conducted using these measures, computed per session and then averaged for each user. We defined a session as a set of aggregated logs produced by a single user within a certain period of time which meets the following conditions: (i) the app starts after being closed or (ii) the app is resumed and since the app was paused either: (a) 60 minutes have passed or (b) the user has moved more than 850 meters. Next, we describe the metrics that we used:

Filter Selection: calculated as the number of filters (including bus route and inbound/outbound bus direction) per session that were selected/deselected by the user.

Navigation: computed as the number of both scrolling and click actions per session.

Days in the Study: measured as the difference between timestamps of the last and first session per user. This metric was used in our abandonment (survival) analysis.

Timestep: calculated as a consecutive time interval of 24 hours where all session values are aggregated and averaged. By "consecutive" we mean that there are no gaps between timesteps since we are doing a relative (rather than absolute) date analysis. For instance, if the user interacted with the app during three sessions (at different times of the day) on 10/12/2019 and then two sessions on 10/17/2019, then timestep $t_{(n)}$ would serve as a time window to average all the values (e.g., number of selected filters) for the three sessions that occurred on 10/12/2019, and timestep $t_{(n+1)}$ would average all the values for the two sessions that occurred the next available date (10/17/2019), and so forth.

System performance: computed as the absolute mean percentage error (MAPE) of the ML model at every timestep. System performance was labeled as "Good" if the error in timestep $t_{(n)} < t_{(n-1)}$, and otherwise labeled as "Bad". We decided to compare timesteps n and $n-1$ instead of using a percentage error threshold because we wanted to accurately account for even the most minimal changes to system performance since regression curves tend to cross at multiple points as we will see later on section 4.1.

4 FINDINGS

After cleaning the data, there were a total of 2,616 participants that installed and used our app, with a balanced group split of 49.8% (1,302) participants in the non-adaptive condition (NAC) and 50.2% (1,314) participants in the adaptive condition (AC). In total we logged 130,939 user sessions, though participants in the AC reported more sessions (total=68,129, M=51.37, SD=198.24) than participants in the NAC (total=62,732, M=47.74, SD=164.47). Additionally, from the total number of participants, we identified a small, but representative sample of people who used the screen reader in each condition (AC=41, NAC=150) who reported a total of 2,667 sessions in the AC (M=65.23, SD=108.31) and 9,453 sessions in the NAC (M=61.42, SD=89.47). A power analysis confirmed that the sample size for both groups of people who are likely blind were suitable to detect the effect of the hypothesis test at the desired level of significance ($\alpha = .05$).

4.1 On Effort Reduction

In order to conduct the statistical analysis for testing the hypotheses related to effort reduction (in terms of lowering filter selection and navigation over time), we split our data set into multiple subsets of data following certain criteria. For H1 we separated the data into 2 subsets based on the adaptive/non-adaptive condition (AC and NAC); For H1.1 we divided the AC group into 2 subsets based on the ML model's performance (good or bad) and included the NAC subset without modifications; And for H1.2 we split both AC

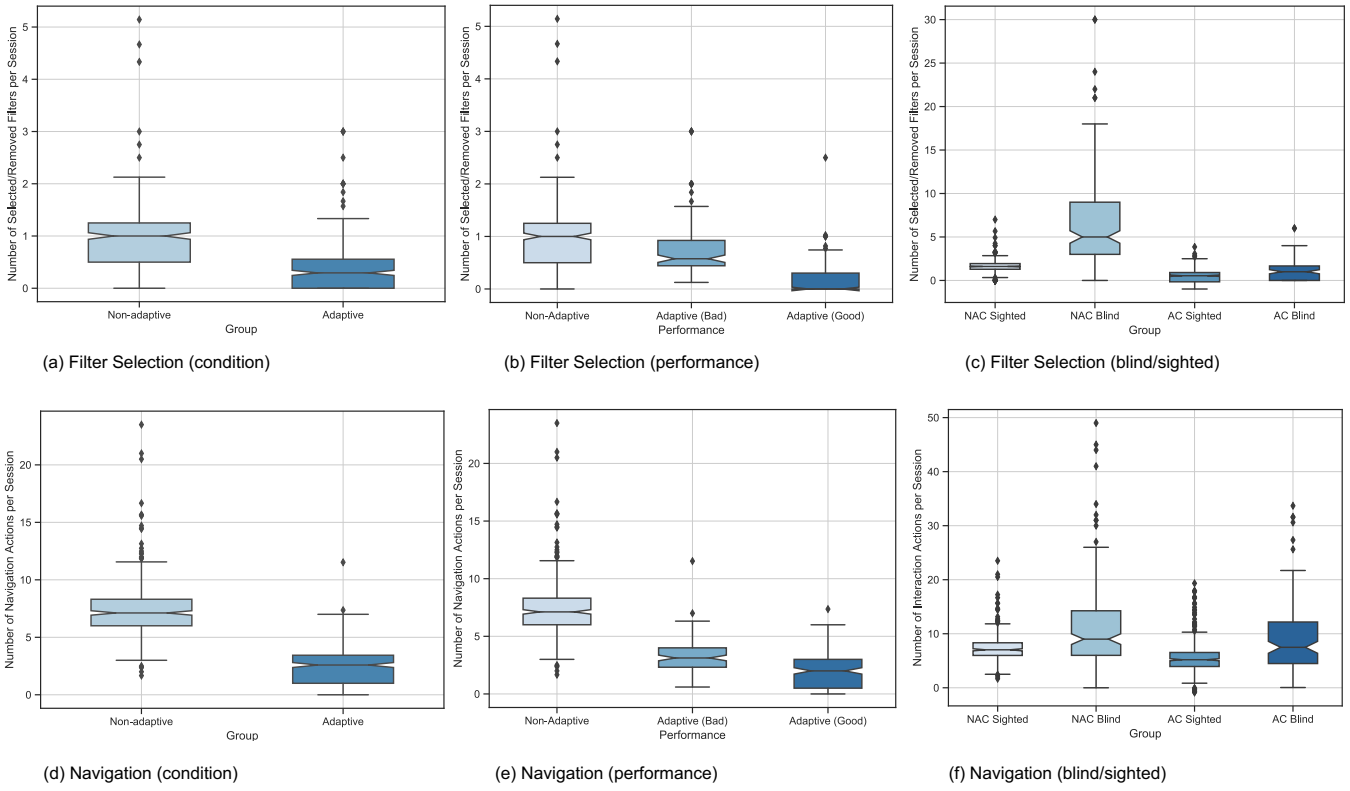


Figure 2: Data distributions for Filter Selection and Navigation features across groups.

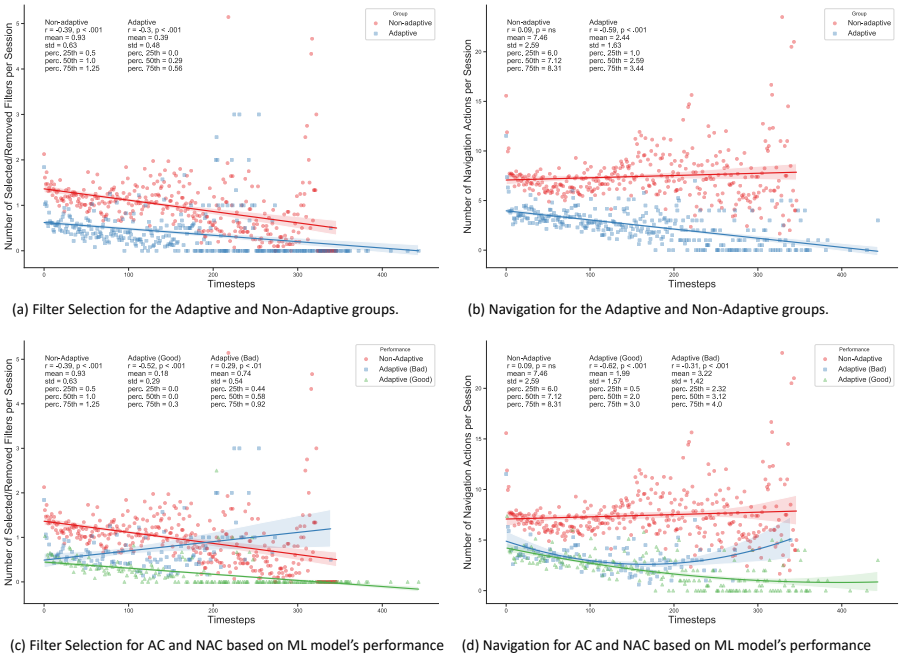
and NAC data sets into two subsets each, that is, blind and sighted participants, by looking at each session log and verifying whether participants used a screen reader.

Figure 2 helps characterize our data distributions. At first glance, we can observe that the notches in the boxplots of Figures 2.a, 2.b, 2.d, and 2.e do not overlap, suggesting that (with 95% of confidence) the true medians for both filter selection and navigation do differ for all groups. For the distribution comparisons between blind vs sighted conditions, we only observed significant differences in filter selection for NAC groups (Figure 2.c) and navigation for AC groups (Figure 2.f). (Detailed statistical results are provided later in this section.) Additionally, it is worth noting that NAC and Blind groups reveal unusual statistical features, such as gaps (e.g., between 24-30 filters for NAC Blind in Figure 2.c) and outliers (e.g., up to 50 navigation actions and 30 filters for the NAC Blind group).

In order to analyze and visualize how patterns of filter selection and navigation changed over time across users in both conditions (H1), we averaged crosswise sessions from all users that occurred at a specific timestep and plotted a single data point for that timestep, and so forth. We obtained 346 timesteps for the NAC and 443 timesteps for the AC. This is the reason why the regression lines for the NAC in Figure 3 seem truncated or shorter than the regression lines for the AC. It is worth highlighting that timesteps are a relative date-based measure, which is different from the number of days users stayed in the study (Section 4.2) or the total number of sessions per group (Section 3.2).

From the scatter plot in Figure 3.a, we can observe that users in AC selected less route filters per session than users in the NAC, though both groups show a trend towards selecting no filters over time. We can also see that NAC regression line is shorter than AC line, indicating that users in the NAC interacted with the app during much fewer timesteps (around 22% less). The mean values and percentiles are much lower for the AC (M=0.39, SD=0.48) than for the NAC (M=0.93, SD=0.63), suggesting that, in general, users in the AC performed less filter selection and therefore experienced less friction (and less effort) while interacting with the app. However, the slope of the regression line for the NAC (r=-0.39, $p < 0.001$) is slightly steeper, possibly indicating that users in this condition could drop filter selection faster than users in the AC over time. If we prolong the NAC regression line we would see that users in this condition would select no filters after around 600 timesteps and both regression lines would converge.

In Figure 3.b we can observe an opposite effect where the number of navigation actions (i.e., scrolls, clicks, etc.) tend to increase over time for users in the NAC (M=7.46, SD=2.59), while decreasing for users in AC (M=2.44, SD=1.63). Both regression lines diverge from each other rather than eventually converge, as we did observe for filter selection (Figure 3.a). This again confirms that users in AC experienced less friction and effort than NAC once the ML model had learned their behavior and adapted the interface. After splitting the AC group to analyze how users behave as a consequence of either good (AC_G) or bad (AC_B) ML model performance, we discovered



(a) Filter Selection for the Adaptive and Non-Adaptive groups.

(b) Navigation for the Adaptive and Non-Adaptive groups.

(c) Filter Selection for AC and NAC based on ML model's performance

(d) Navigation for AC and NAC based on ML model's performance

Figure 3: Interaction characterization over time.

different user behavior patterns for filter selection and navigation. We found an inverse correlation between the ML model's performance and the users' behavior for filter selection and navigation. I.e., the better the model performed, the less users selected filters and navigated, and conversely, the worse the model performed the more users selected filters and navigated.

Although AC_B users who experienced a bad system performance (i.e., the ML model made incorrect predictions that did not adapt to their needs) made fewer filter selection actions on average (M=0.74, SD=0.54) than NAC (M=0.93, SD=0.63), they demonstrated a marked trend towards increasing the number of route filtering and navigation actions as time went on (Figure 3.c). During the first 200 timesteps, AC_B users made less effort in terms of filter selection, even when the model performed badly, in comparison with users in the NAC where no predictions were available. However, filter selection gradually increased to the point that, around timestep 200, both regression lines crossed each other, indicating that users in the NAC started selecting fewer filters than users in AC_B. We noticed a similar effect on navigation where, although the AC_B subgroup performed fewer navigation actions (M=3.22, SD=1.42) than NAC (M=7.46, SD=2.59), there was a trend towards increasing the number of navigation actions per session. This suggests that a poor adaptation may lead to increasing levels of friction, effort, and possibly frustration, thereby affecting UX.

To determine whether there were statistically significant differences between groups, we examined their characteristics and drew insights on what kind of navigation and filter selection patterns could be observed for different types of users. Note that we report only statistically significant findings at $p < .05$, $p < .01$, and $p < .001$.

For data that met the parametric assumptions we applied a one-way independent ANOVA test, whereas for all other cases we used the Kruskal-Wallis test. If a significant difference was found, we performed follow up pairwise comparisons using the independent t-test or the Mann-Whitney test, respectively, with Holm-Bonferroni correction. Effect sizes are reported using Cohen's d.

In Table 1 we summarize the statistical analysis for all the individual interactions logged by our system. These individual interactions were grouped into 3 categories: navigation (all button clicks and scrolling actions), filter selection (select and remove bus routes and trip direction filters), and the number of results shown and predicted by the ML model (show IN/OUT routes). Additionally, we report comparative statistics for aggregated interactions using the 3 categories aforementioned, as shown in Table 2. Five groups are analyzed: AC , NAC, AC^+ (AC with screen reader ON), NAC^- (NAC with screen reader OFF), and AC_B (AC who experienced bad system performance).

Overall, in Table 1, we only see statistically significant differences on a small subset of individual interactions. However, when conducting a one-tailed test (t) for each individual interaction reported as a significantly different, we realize that samples did not always fall into the same side of the critical area as the rest of interactions in the same category. For instance, category navigation for AC^+ vs NAC^- had 6 one-tailed results that fell into NAC^- and 2 results that fell into AC^+. Therefore, we cannot use these results to accept or reject our hypotheses, however, these results allow us to get a clearer idea about what kind of specific interactions users from each group encountered the most friction and effort. Additionally, the results shown in Table 1 help us identify patterns in user behavior that support our initial assumptions, e.g. by looking at the

Table 1: Summary of the statistical analysis differentiated by Interaction Type. Five groups are analyzed: AC , NAC, AC^+ (AC with screen reader ON), NAC^- (NAC with screen reader OFF), and AC_B (AC who experienced bad system performance). Descriptive Statistics ($\overline{X}$, σ) are shown from column AC to column AC_B. Statistical Tests are shown from column AC vs. NAC to column AC^+ vs. NAC^-. We report the statistic value (t-test or Mann-Whitney) with significance ***p < 0.001, **p < 0.01, *p < 0.05. Effect size assessed via Cohen's d (d). One-tailed test (t) shows which group's critical area the test statistic falls into.

	Within Groups										Between Groups								
	AC		NAC		AC^+		NAC^-		AC_B		AC vs. NAC			AC_B vs. NAC			AC^+ vs. NAC^-		
Interaction Type	$\overline{X}$	σ	$\overline{X}$	σ	$\overline{X}$	σ	$\overline{X}$	σ	$\overline{X}$	σ	*Stat*	*d*	*t*	*Stat*	*d*	*t*	*Stat*	*d*	*t*
Navigation																			
I01: click (alert_dismiss)	.004	(.181)	.003	(.063)	0	(0)	0	(0)	.004	(.267)	ns			ns			ns		
I02: click (back)	.269	(.907)	.46	(1.42)	.18	(.401)	.561	(1.19)	.441	(1.03)	ns			ns			ns		
I03: click (location_bar)	.216	(.933)	.473	(1.67)	.133	(.340)	.366	(.877)	4.56	(.792)	ns			ns			ns		
I04: click (map_stop)	.074	(.543)	.126	(1.14)	.013	(.163)	.146	(.646)	.119	(.238)	.03***	.11	NAC	ns			.01***	.4	NAC^-
I05: click (map_user_loc)	.037	(.407)	.024	(.398)	0	(0)	.073	(.463)	.039	(.563)	ns			ns			.62***	.34	NAC^-
I06: click (route_page)	.316	(.929)	.526	(1.24)	.683	(.412)	.173	(1.09)	.502	(.763)	ns			ns			1.23*	.81	AC^+
I07: click (route_tab)	.339	(1.01)	.553	(1.34)	.707	(.417)	.182	(1.17)	.623	(.661)	ns			1.7***	.83	AC_B	.06*	.8	AC^+
I08: click (schedule_tab)	.113	(.575)	.144	(.477)	0	(0)	.122	(.327)	.238	(.265)	.03**	.2	NAC	.72	.78	AC_B	.09*	.8	NAC^-
I09: click (stop_search)	.057	(.294)	.022	(.205)	.067	(.275)	0	(0)	.031	(.355)	ns			ns			ns		
I10: click (stop_page)	.049	(.253)	.091	(.435)	.113	(.317)	.122	(.395)	.087	(.428)	ns			ns			ns		
I11: click (toggle_map)	.327	(1.05)	.225	(.921)	.366	(.291)	.093	(.574)	.258	(.603)	.03**	.23	AC	ns			.01*	.73	AC^+
I12: scroll (route_filter)	.551	(2.17)	.098	(3.48)	1.78	(10.5)	.512	(2.27)	.077	(.709)	.03***	.42	AC	.12***	.13	NAC	ns		
I13: scroll (schedule_list)	2.656	(7.57)	4.216	(8.94)	.853	(4.13)	15.341	(76.2)	4.02	(6.23)	.04*	.24	NAC	ns			1.52*	.81	NAC^-
I14: scroll (stop)	.003	(.074)	.016	(.526)	0	(0)	0	(0)	.014	(.493)	ns			ns			ns		
I15: scroll (stop_list)	.858	(6.23)	1.807	(14.1)	.233	(.667)	2.659	(6.85)	1.92	(.526)	ns			ns			.87**	.75	NAC^-
Filter Selection																			
I18: select route	.083	(1.05)	.141	(1.78)	0	(0)	.049	(.309)	.178	(.112)	.03***	.08	NAC	1.82	.63	AC_B	.03***	.34	NAC^-
I19: select filter (IN)	.161	(.582)	.186	(.519)	.16	(.463)	.146	(.353)	1.89	(.788)	ns			ns			ns		
I20: select filter (OUT)	.121	(.413)	.186	(.517)	.367	(.547)	.024	(.154)	.154	(.339)	ns			.19***	.31	NAC	ns		
I21: remove filter (IN)	.155	(.565)	.071	(.369)	.047	(.240)	0	(0)	.073	(.342)	.03**	.19	AC	ns			ns		
I22: remove filter (OUT)	.142	(.440)	.076	(.389)	.14	(.366)	0	(0)	.092	(.659)	ns			ns			ns		
Shown Results																			
I23: show route (IN)	.073	(.311)	.359	(.588)	.498	(.268)	.042	(.970)	.329	(.213)	.04*	.62	NAC	ns			.49***	.34	AC^+
I24: show route (OUT)	.308	(.299)	.063	(.557)	.023	(.613)	.538	(.751)	.278	(.799)	.04*	.56	AC	ns			.36***	.56	NAC^-

intersection of rows I04 and I05 (navigational interactions with the city map), and the last column (screen reader vs. non-screen reader users), we can support the assumption that screen reader users are low vision or blind since they did not interact with visual content.

It is worth noting that when we aggregated those individual interactions into their corresponding categories, we observed more statistical power and larger effect size (Table 2). Also, we note that some of the trends for statistical analysis of several individual interactions reverse for the category they belong to. For instance, there was a trend towards AC_B performing more navigation actions than NAC (Table 1), but this trend reversed to NAC when interactions were aggregated (Table 2) – Simpson's paradox.

Testing H1 - AC vs. NAC: in average, participants in AC reported more navigation on map (M_{I05}=.03, M_{I11}=.32) and bus stop features (M_{I09}=.05) than NAC (M_{I05}=.02, M_{I11}=.22, M_{I09}=.02), while NAC reported more scrolling on bus schedule (M_{I13}=4.216) than AC (M_{I13}=2.656). Likewise, participants in AC performed more filter selection actions (M_{I21}=.155, M_{I22}=.142) than NAC (M_{I21}=.07, M_{I22}=.076). Both filter selection and navigation showed statistically significant differences, but with a small effect size ($.03 \geq d \leq .22$). However, from the aggregated data, it can be concluded that both navigation and filter section effort in the NAC group was statistically significantly higher than the AC group (Mann–Whitney U=43.5, p=.002) and (U=59.3, p < .001), respectively. Additionally, a Mann-Whitney test indicated that number of results shown per session was significantly greater for the NAC group (M=17.3, SD=4.3) than for the AC group (M=5.2, SD=2.3, U=16.1, p=0.003), ratifying that users in this condition had to make extra effort to access and navigate the information. Effect size is large for the three tests ($.63 \geq d \leq .87$). H1 was confirmed.

Table 2: Aggregated statistical analysis. ***p < 0.001, **p < 0.01, *p < 0.05. d = Cohen's d. t = one-tailed test.

Pair-wise Comparison	Navigation (Stat, d) t	Filter Selection (Stat, d) t	Shown Filters (Stat, d) t
AC vs. NAC	(43.5**, .87) NAC	(59.3***, .81) NAC	(16.1**, .63) NAC
AC_B vs. NAC	(28.3***, .65) NAC	(11.7***, .78) AC_B	(35.1**, .48) NAC
AC^+ vs. NAC^-	(42.9*, .59) AC^+	(19.6***, 0.89) NAC^-	(39.9***, 0.91) NAC^-
AC^+ vs. AC^-	(29.8**, .66) AC^+	ns	ns
NAC^+ vs. NAC^-	(13.9**, .70) NAC^+	(32.1***, .88) NAC^+	ns

Testing H1.1 - AC_B vs. NAC: from the individual interactions analysis we found that AC_B's effort was significantly higher for actions such as looking at route (M_{I07}=.62, SD_{I07}=.66), schedule (M_{I08}=.23, SD_{I08}=.26) and route filter selection (M_{I18}=.17, SD_{I18}=.11) than the NAC group (M_{I07}=.55, SD_{I07}=1.34), (M_{I08}=.14, SD_{I08}=.47), (M_{I18}=.14, SD_{I18}= 1.78), respectively.

The NAC group was significantly higher for scrolling route information (M_{I12}=.09, SD_{I12}=3.48) and outbound filter selection (M_{I20}=.18, SD_{I20}=.51) than AC group (M_{I12}=.77, SD_{I12}=.07), (M_{I20}= .15, SD_{I20}=.33), respectively. From the aggregated data, a Mann-Whitney test indicated that the amount of navigation was statistically significantly higher (U=28.3, p=0.004) than the AC group,

whereas an opposite effect was noted for the number of selected filters per session where the AC group was statistically significantly higher than the NAC group (U=11.7, p < 0.001) with a large effect size ($.65 \geq d \leq .78$). H1.1. was not confirmed.

Testing H1.2 - AC^+ vs. NAC^-: overall, users in the NAC^- interacted significantly more times with the map (e.g., M_{I04}=.14, SD_{I04}=.64), schedule information (e.g., M_{I13}=15.34, SD_{I13}=76.2), and the list of bus stops (M_{I15}=2.65, SD_{I13}=6.85), than users in the AC^+ (M_{I04}=.01, SD_{I04}=.16), (M_{I13}=.85, SD_{I13}=4.1), (M_{I15}=.23, SD_{I13}=.66). On the other hand, users in the AC^+ navigated significantly more through the route information (M_{I06}=.68, SD_{I06}=.41) than users in the NAC^- (M_{I06}=.17, SD_{I06}=1.09), with a relatively large effect size. From the aggregated data we can conclude that users in the NAC^- make statistically significantly more effort than blind users in the AC^+ in terms of filter selection (U=19.6, p < .001), while users in the AC^+ make significantly more navigation effort than users in the NAC^- (U=42.9, p=.03), with a large effect size ($.59 \geq d \leq .91$). H1.2 was confirmed.

Additionally, separated statistical analysis on groups of both blind and sighted users and found that blind users in the AC^+ had significantly more navigation effort (M=7.36, SD=3.8) than sighted users in the AC^- (M=6.18, SD=1.2, U=29.8, p=.008), while no statistically significant difference was observed for filter selection. On the hand, blind users in NAC^+ had significantly more filter selection (M=5.1, SD=5.6) and navigation efforts (M=9.22, SD=4.3) than sighted users in the NAC^- (M=2.39, SD=1.4, U=32.1, p < .001) (M=8.01, SD=2.1, U=13.9, p=.006), respectively.

4.2 On App Abandonment Rate

We analyzed the rate of app abandonment for participants in each condition via a survival analysis. For the purposes of censoring, we established a 500-day window and made observations of participants who were subject to the abandonment event as well as participants who had not been subject to it yet (labeled as a right-censored). I.e., participants that we cannot determine whether they will abandon or when they will. We continued tracking devices after the time window (the remaining 50 days for a total of 550 days or ≈18 months), so the right-censored data we collected allowed us to properly account for participants who continued using the app beyond the end of our data sample in survival analyses.

We calculated the total duration that each participant had been using the app before abandonment (time to event) and found that participants in the AC (M=118.6, SD=173.4) demonstrated slightly higher rate of abandonment than subjects in NAC (M=125.31, SD=173.2). The 50^{th} and 75^{th} percentiles confirmed this minor difference for the abandonment rate, being 16.5 days and 175.7 days for the AC and 23.2 days and 201 days for NAC, respectively. After having analyzed the shape of the distributions for both AC and NAC, we realized that they both resemble a chi-square distribution, thus we can assume that the participants follow a predictable pattern and therefore a parametric survival analysis can be performed. We examined survival analysis using a Weibull distribution.

Figure 4 depicts the survival probability estimated using the Weibull test. The plot suggests there is no significantly different abandonment rate between participants in both conditions since the confidence intervals ($\alpha = .001$) overlap throughout the timeline.

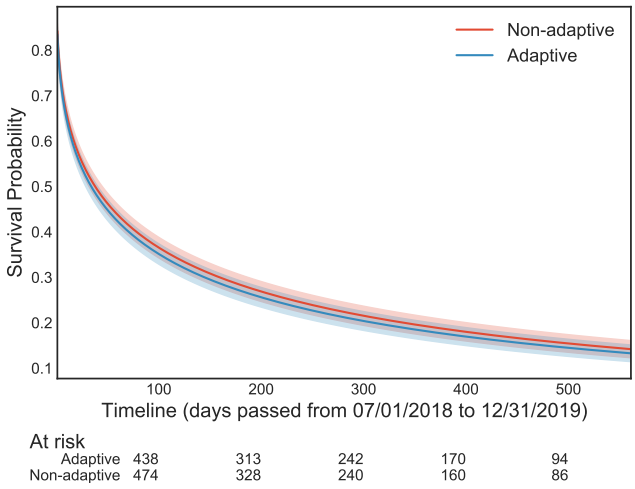


Figure 4: Weibull Survival Analysis.

The Weibull test showed no statistically significant difference between conditions (X^2=13.4, DF=2, p=0.31). Also, we calculated the number of participants at risk (users who have not had an abandonment event before time t) at intervals of 100 days and found that participants in AC dropped out 3% faster than NAC in the first 100 days, while roughly abandoning at the same rate than NAC in the next ≈450 days. This trend can also be seen in the number of participants who continued using the system after the 500-day window (right-censored), with NAC retaining 86 censored devices while AC retaining 94. To compare both survival distributions, we ran both a Log-Rank test (X^2=0.44, DF=1, p=0.51) and a Wilcoxon test (X^2=31.38, DF=1, p=0.86). Since there was no statistically significant difference, H2 was not confirmed.

5 DISCUSSION

In this work, we conducted a long-term assessment and an A/B study during a 18-month window with 2,616 users who voluntarily downloaded our app. In contrast to previous studies, the amount of information logged throughout this period of time is far greater than most studies on adaptive interfaces. However, this scale comes with some limitations and unexpected effects, as discussed below.

Long-term assessment limitations: A portion of the sampled users in each group may have switched between adaptive and non-adaptive conditions. If a user either re-installed the app on the same phone or installed the app on a new phone, they would be randomly assigned to a condition. For privacy reasons, we did not collect user demographics nor store personal information, making tracking of a person by condition impossible. While this effect was probably limited and balanced, it might have biased the data towards a particular group and hindered the traceability of user's interaction in the long term for comparison and adaptation purposes.

Challenges on labeling: The way our adaptive mobile interface organically collected labels for transit behaviors and trained our ML algorithm likely reduced effort and friction since users were not explicitly queried to provide any additional feature information for the adaptation. While being an inexpensive manner to label large amounts of data without compromising user experience, this method still had some challenges related to label accuracy that

needed to be addressed. Specifically, we applied a set of heuristics to help us infer the user's true intention in cases when they selected or deselected the same route filter back and forth several times in a certain period of time (due to curiosity, inexperience with the app, mistakes, etc.). We need more sophisticated methods to deal with this kind of behavior. Our interaction design does not require users to share the route and direction they ended up taking as feedback for labeling and measuring performance. This information is difficult to motivate users to share, as they receive no immediate benefit from the action. In addition, users might make other choices, such as not traveling or using an alternative mode, such as taking an Uber. Instead, we relied on observable user's filtering behaviors (selection/deselection). For this reason, capturing the users' explicit desire remains a challenging problem.

Impact on long-term adaption: Long-term field studies expose research to additional effects, such as the impact of user adaptation and rate of abandonment. One of these effects is that, although users in the NAC never received predictions, they did show a gradual drop in filter selection effort. There are some possible reasons that explain this pattern. (i) In early timesteps, there was a considerable amount of new users in both conditions who were not familiar with the app who, therefore, spent more time exploring the app's features (meaning more navigation and filter selection). Regardless of whether users received predictions or not, a reduction in filter selection would be expected as they became accustomed to app features. (ii) Information shown in the app was context-sensitive, so even though users in the NAC did not receive predictions, users in less dense regions of the city likely found what they needed in the first few rows of the shown results (i.e., above the fold), thereby removing the need for sighted users to filter. (iii) Users in the NAC may have stopped selecting filters because they realized that their route preferences were not stored or 'learned' by the system. Since selecting filters was extra effort, they may have been unwilling to keep filtering over the long term. Instead, they may have preferred to scroll down instead or just wait for their preferred bus to appear in the first rows of the results. This would lead to more navigation and less filter selection (as shown in Figure 3).

Another effect that is worth highlighting is the increasing amount of effort that users in the AC had to make when the ML produced incorrect predictions. This possibility raises several unresolved questions, such as: why did effort not remain constant over time for those users? what factors made them increasingly select/deselect more filters? We speculate on several rationales for this: (i) Since the number of active sessions dropped significantly after around timestep 150, beyond that point the mean number of selected filters per timestep is highly influenced by outliers. (ii) Users in the AC continued selecting filters when the system performed badly because, unlike users in the NAC who never received either good or bad predictions, AC users had previously received "good" predictions from the system. They (implicitly) might have known that their bus route preferences were somehow stored or "learned" by the system, so they kept trying and did not give up in the hope that the system would make a good prediction again in the near future (a trust bias). (iii) We looked into the log traces of some users in the AC and realized that sometimes the system insisted on showing a particular bus route that the user was not interested in at that time (maybe they were interested in that route int the past, so the system learned to predict it), forcing the user to repeatedly deselect the same filter during the same session. That can be considered as the "tyranny of adaptation". The issue arises from the lack of evidence of recent activity of the user as part of the model. That is, if the model understood that the user has deselected a filter in the last few minutes, the model could adjust to this situation correctly. Note that the appearance of this problem is due to the system's deployment into the real-world. This kind of interaction effect would not usually show up in a laboratory environment.

Impact on screen reader users: Interestingly, our study showed that our adaptive interface significantly reduced navigation and filter selection efforts for users both with and without visual impairments. This achievement is relevant because interactions with smartphones are particularly time and effort consuming, especially for people with disabilities, due to smaller screen sizes and input options. We anticipate that this reduction in effort can lower cognitive and interaction load, which is especially important for people with other kinds of disabilities such as cognitive or dexterity disabilities. Since the UX community is still investigating how to make adaptive interactions effective [26], there is an opportunity to embrace universal design and create adaptive experiences that work for all users. It is important to take this step before the UX community settles on standardized patterns that are difficult to make accessible.

ML model's performance: From a confusion matrix analysis we got the following statistics: true positives (filters were predicted but not removed, TP=562,404), true negatives (filters were neither predicted nor selected, TN=1,736,324), false positive (predicted filters were removed, FP=167,991), and false negatives (additional filters were selected, FN=302,833). Though our model got an accuracy of 83%, we did not rely on this performance measure because our dataset was unbalanced and asymmetric, instead, we relied on precision, recall and F1 score. Our ML model reported modest success that could be improved. For example, merely repeating the last filters selected obtains precision and recall rates of 56%. In contrast, analyses using our ML model had a precision of 77% and recall of 65%. We have reached recall rates in the 90% range for certain combinations of features and heuristics, but these are usually paired with very low precision.

Abandonment rate: Surprisingly, we did not observe a statistically significant difference for the abandonment rate between both AC and NAC groups. They both seem to abandon use of the app at very consistent, similar rates. We have identified possible reasons for this observed effect. One reason is that there may be a noise factor where people left the city, or transit, which manifested in the data as abandonment (e.g., job change, degree completion, etc.). Also, participants might have switched between groups since the AC/NAC condition is assigned randomly when the app is reinstalled or installed in a new device. Thus, some abandonment may have actually been a switch to another group.

Additionally, our ML model uses simple features, but modeling travel patterns and human behavior can be much more complex, encompassing changes in lifestyle and routines (e.g., starting working out at the gym, a new job, etc.), occasional changes in trip plans (e.g., a doctor's appointment), changes in social circles (e.g., new friends, hobbies, activities), multi-scale time sensitivity (e.g., fine-grained sensitivity as a minute, or coarse-grained sensitivity as academic calendars), and more. This is a lesson for UX designers

and practitioners: the community needs to think better about the dimensionality of the learning to avoid adverse effects of adaptation, such as user frustration and app abandonment.

On the other hand, though the survival analysis produces reliable results for predicting app abandonment, interpreting it in isolation does not tell us much about the quality of both the adaptation and the user experience. Therefore, it should be considered in combination with other metrics. For instance, although both groups abandon at the same rate, the AC group reported significantly more sessions and timesteps than NAC. This means that, while a user in the NAC who opened and used the app every 3 months and did not abandon the app after 1 year (for a total of only 4 sessions), another user in the AC abandoned after 10 months but interacted with the app for more than 500 sessions. The higher session result implies that adaptation has long term value.

Finally, if users in the AC who experienced a bad system performance had more interaction effort than users in the NAC, why did they not abandon the app earlier? A possible reason is that the user experience is not necessarily black-or-white. I.e., since the adaptive interface interleaves bad predictions with good predictions and all results are time and location based, then the user experience is not totally unpleasant. This effect can be explained by the notion of 'fail-soft' [21], where inference or prediction errors still move users closer to their goal while providing partial operational capability.

6 RELATED WORK

Adaptive UIs can be classified into two categories, informative and generative [20]. Informative interfaces use filtering and sorting functionality to offer users with recommendations while Generative interfaces focus on creating new knowledge. Examples of Informative interfaces are Amazon and Netflix recommendation systems, as well as our own interface. Examples of Generative Interfaces are Gmail's Smart Compose feature. In both instances, these interfaces automatically adjust content, layout, or visual presentation based on detected changes in device, context, task, or demands. Adaptive UIs are particularly powerful in tackling information overload [3, 17], increased task complexity [22, 23], and inclusive design [10, 24, 25]. Mobile UI particularly benefits from adaptations that reduce navigation and selection efforts because (i) mobile tasks and contexts vary greatly, (ii) small screens limit interaction and content space, and (iii) the on-the-go usage often limits attention, making every second count [26].

Interestingly, in UI design practice, the function of adaptive UIs is still a heavily contested space. Systems such as SUPPLE/ARNAULD [11, 12] and SUPPLE++ [16] have shown evidence that adaptive UIs significantly improve the accuracy, speed and satisfaction of users with motor impairments compared to default systems [15]. Users with disabilities often pay a higher cost to navigate UIs in terms of cognitive and physical effort. Adaptive UIs can also be used to democratise digital access by overcoming implicit biases that are built into most systems [13]. However, there are also risks. The classic example of negative experience with adaptive user interfaces is the Microsoft Adaptive Menus [7]. These negative experiences are related to a lack of control of adaptive process and difficulty in prediction system changes [7, 9, 18]. In the case of adaptive menus, menu items are not located where the user expects them to be, leading to frustration [5] and negatively impacting user performance when asked to complete new tasks [8].

These mixed evaluations have led to more researchers exploring the factors necessary for successful development of adaptive user interfaces. Gajos et al. [14] identified three factors that significantly influence user experience: (i) accuracy of adaptation, which influences user performance, (ii) frequency of adaptation, which influences user perception on the costs and benefits of adaptation, and (iii) the complexity of the task and the frequency of interaction with the interface. These findings were echoed in subsequent HCI research [2, 5, 6] as well in our own work: (i) for our long-term assessment, we evaluated how the user's performance was influenced in terms of navigation and interaction effort as a result of variations in the accuracy of the prediction model; (ii) we observed that at least 50% of the population abandoned during the first 25 days of the assessment, and we conjecture that a likely reason for it was that the users did not perceive adaptation during this time window due to the system required at least 80 samples to be able to make predictions, however, those users who kept using the app for a longer period perceived continuous adaptations based on changes on their transportation patterns; and (iii) we observed that those users who used the app regularly received more accurate predictions and therefore demonstrated a significant reduction on navigation effort as a result of adaptation over time.

7 CONCLUSIONS

In this work, we consider a long-term assessment of adaptation, in comparison to other studies, of a mobile app that collapses interactions by learning and pre-filtering the information shown to users. We provided empirical evidence that a well-performing adaptive interface leads to reduced interaction effort, while poor performance may lead to more interaction effort than no adaptation. This *tyranny of adaptation* is due to a mismatch between the machine learning model and the contextual environment of real-world use. We also provide evidence that adaptive UIs appear to reduce interaction efforts for users who are blind or low vision and highlight further opportunities in this area. Our survival analysis demonstrates that users in both adaptive and non-adaptive conditions abandon at a very similar rate, and we provide possible explanations for this effect. Finally, some of the most remarkable lesson learned from this work is that in order to build better adaptive UI, more robust and self-reported information and feedback from users is needed so, i) ML models and other adaptation mechanisms can be adjusted opportunely to improve their performance, and ii) a deeper understanding of the reasons why users abandon the app can be acquired. However, this ideal scenario imposes some challenges due to the long-term nature of this assessment because users rarely are motivated to provide this feedback over long periods of time, so we still need to conduct further research in this area.

ACKNOWLEDGMENTS

The contents of this paper were developed under grants from the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant numbers 90RE5011 and 90REGE0007). We would like to thank everyone who has worked on Tiramisu over the years and the Port Authority of Allegheny County for their insights and assistance.

REFERENCES

[1] L. Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32.

[2] Robert Bridle and Eric McCreath. 2006. Inducing shortcuts on a mobile phone interface. In *Proceedings of the 11th international conference on Intelligent user interfaces*. Association for Computing Machinery, New York, 327–329.

[3] P Brusilovsky. 2001. Adaptive hypermedia, user modeling and user-adapted. *Interaction Journal* 11, 1-2 (2001), 87–110.

[4] Rich Caruana, Nikos Karampatziakis, and Ainur Yessenalina. 2008. An empirical evaluation of supervised learning in high dimensions. In *In International Conference on Machine Learning (ICML*. Association for Computing Machinery, New York, NY, USA, 96–103.

[5] Andy Cockburn, Carl Gutwin, and Saul Greenberg. 2007. A predictive model of menu performance. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. Association for Computing Machinery, New York, 627–636.

[6] Tilman Deuschel. 2018. On the Influence of Human Factors in Adaptive User Interface Design. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*. Association for Computing Machinery, New York, 187–190.

[7] Leah Findlater and Krzysztof Z Gajos. 2009. Design space and evaluation challenges of adaptive graphical user interfaces. *AI Magazine* 30, 4 (2009), 68–68.

[8] Leah Findlater, Karyn Moffatt, Joanna McGrenere, and Jessica Dawson. 2009. Ephemeral adaptation: The use of gradual onset to improve menu selection performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, 1655–1664.

[9] Leah K Findlater. 2004. *Comparing static, adaptable, and adaptive menus*. Ph.D. Dissertation. University of British Columbia.

[10] Josef Fink, Alfred Kobsa, and Andreas Nill. 1998. Adaptable and adaptive information provision for all users, including disabled and elderly people. *New review of Hypermedia and Multimedia* 4, 1 (1998), 163–188.

[11] Krzysztof Gajos and Daniel S Weld. 2004. SUPPLE: automatically generating user interfaces. In *Proceedings of the 9th international conference on Intelligent user interfaces*. Association for Computing Machinery, New York, 93–100.

[12] Krzysztof Gajos and Daniel S Weld. 2005. Preference elicitation for interface optimization. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*. Association for Computing Machinery, New York, 173–182.

[13] Krzysztof Z Gajos. 2014. Making the web more inclusive with adaptive user interfaces. In *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems*. Association for Computing Machinery, New York, 1–1.

[14] Krzysztof Z Gajos, Mary Czerwinski, Desney S Tan, and Daniel S Weld. 2006. Exploring the design space for adaptive graphical user interfaces. In *Proceedings of the working conference on Advanced visual interfaces*. Association for Computing Machinery, New York, 201–208.

[15] Krzysztof Z Gajos, Daniel S Weld, and Jacob O Wobbrock. 2008. Decision-Theoretic User Interface Generation. In *AAAI*, Vol. 8. AAAI Press, New York, 1532–1536.

[16] Krzysztof Z Gajos, Jacob O Wobbrock, and Daniel S Weld. 2007. Automatically generating user interfaces adapted to users' motor and vision capabilities. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*. Association for Computing Machinery, New York, 231–240.

[17] Kristina Höök. 1998. Evaluating the utility and usability of an adaptive hypermedia system. *Knowledge-Based Systems* 10, 5 (1998), 311–319.

[18] Kristina Höök. 2000. Steps to take before intelligent user interfaces become real. *Interacting with computers* 12, 4 (2000), 409–426.

[19] Anthony Jameson. 2007. Adaptive interfaces and agents. In *The human-computer interaction handbook*. CRC Press, New York, 459–484.

[20] Pat Langley. 1997. Machine learning for adaptive user interfaces. In *Annual Conference on Artificial Intelligence*. Springer, New York, 53–62.

[21] Henry Lieberman, Hugo Liu, Push Singh, and Barbara Barry. 2004. Beating Commons Sense into Interactive Applications. *AI Magazine* 25 (2004), 63–76.

[22] Joanna McGrenere, Ronald Baecker, and Kellogg Booth. 2002. *The design and evaluation of multiple interfaces: A solution for complex software*. University of Toronto Canada, Toronto.

[23] AF Norcio and J Stanley. 1988. *Adaptive human-computer interfaces*. Technical Report. NAVAL RESEARCH LAB WASHINGTON DC.

[24] Constantine Stephanidis, Alex Paramythis, Michael Sfyrakis, A Stergiou, Napoleon Maou, A Leventis, George Paparoulis, and Charalampos Karagiannidis. 1998. Adaptable and adaptive user interfaces for disabled users in the AVANTI project. In *International Conference on Intelligence in Services and Networks*. Springer, Association for Computing Machinery, New York, 153–166.

[25] Shari Trewin. 2000. Configuration agents, control and privacy. In *Proceedings on the 2000 conference on Universal Usability*. Association for Computing Machinery, New York, 9–16.

[26] Qian Yang, John Zimmerman, Aaron Steinfeld, and Anthony Tomasic. 2016. Planning adaptive mobile experiences when wireframing. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*. Association for Computing Machinery, New York, 565–576.

[27] John Zimmerman, Anthony Tomasic, Isaac Simmons, Ian Hargraves, Ken Mohnkern, Jason Cornwell, and Robert Martin McGuire. 2007. Vio: A Mixed-Initiative Approach to Learning and Automating Procedural Update Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 1445–1454. https://doi.org/10.1145/1240624.1240843