AutoSens: Inferring Latency Sensitivity of User Activity through Natural Experiments

Parth Thakkar*
University of Illinois, Urbana-Champaign
parthdt2@illinois.edu

Rohan Saxena*
Carnegie Mellon University
rohansaxena@cmu.edu

Venkata N. Padmanabhan
Microsoft Research India
padmanab@microsoft.com

ABSTRACT

We consider the problem of inferring the latency sensitivity of user activity in the context of interactive online services. Our method relies on natural experiments, i.e., leveraging the variation in user-experienced latency seen in the normal course. At its core, our technique, dubbed AutoSens, compares the distribution of latency of the user actions actually performed with the underlying distribution of latency independent of whether users choose to perform any action. This then yields a normalized user preference based on latency. We discuss ways of mitigating various confounders and then present our findings in the context of a large online email service, Microsoft Outlook Web Access (OWA).

CCS CONCEPTS

• **General and reference** → **Measurement**; **Metrics**; **Estimation**; **Performance**.

ACM Reference Format:
Parth Thakkar, Rohan Saxena[1], and Venkata N. Padmanabhan. 2021. AutoSens: Inferring Latency Sensitivity of User Activity through Natural Experiments. In *ACM Internet Measurement Conference (IMC '21), November 2–4, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3487552.3487839

1 INTRODUCTION

Latency is a key metric defining the user experience in the context of online services such as email, search, e-commerce, and more. The conventional wisdom is that users react negatively to latency — the higher the latency of a service, the lesser the activity that users are likely to perform in the service, which in turn means lost revenue for the (commercial) service. For instance, there have been studies from Amazon [19] (a 100 ms increase in latency can cause a drop of 1% in sales), Google [14] (half a second increase in latency results in a 20% drop in traffic), and Akamai [31] (a 100 ms increase in latency can cause a drop of 7% in conversion).

Studies such as the ones above are typically performed through active intervention in the form of A/B tests. For instance, in the case of the Amazon study, an additional 100 ms of latency was introduced for the treatment group, while in the case of the Google study, the treatment involved increasing the number of search results returned to 20 or 30 instead of the default 10, and the increase in latency was an intended side-effect, which nevertheless impacted user activity.

In this paper, we present AutoSens, a general framework for automatically inferring the latency sensitivity of user activity through *natural experiments*. Specifically, we leverage the variation in latency that occurs in the normal course to glean the impact that latency has on the level of user activity. Compared to the prior work noted above, AutoSens is easier to use (avoiding the complexity of active intervention) and also safer (no risk of adverse impact on user experience).

The approach taken by AutoSens is to compare the *biased* distribution of latency (i.e., the distribution of latency of user actions actually observed, which would reflect the impact, if any, of latency on user activity) with the *unbiased* distribution reflecting the underlying latency of actions without regard to when user actions are actually performed. If the biased distribution is shifted to the left (towards lower latency) compared to the unbiased distribution, that would indicate a greater likelihood of user activity when the latency is lower. We turn this basic insight into a metric dubbed `normalized latency preference`, which quantifies the relative likelihood of user activity at different levels of latency. Applying this methodology, however, requires mitigating confounding factors such as time-of-day effects, which we consider.

We apply AutoSens in the context of Microsoft Outlook Web Access (OWA), a large web-based email service that serves millions of users globally. We evaluate AutoSens for a variety of slices of the data encompassing various types of user actions, different classes of users (paying versus free tier), user conditioning based on experience, and more. Our findings broadly accord with intuition. [1]

2 AUTOSENS DESIGN

2.1 Overview

AutoSens relies on minimal telemetry — logs of user actions, including timestamp, as recorded at the server (e.g., web access logs). Specifically, for our analysis, we need tuples of the form (T, A, L, M) for every action A (e.g., selecting a mail item, switching between mail folders, etc.) started at time T, and which had an end-to-end latency L. The end-to-end latency L is measured by the client (e.g.,

*Work done while these authors were Research Fellows at Microsoft Research India.

IMC '21, November 2–4, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-9129-0/21/11…$15.00
https://doi.org/10.1145/3487552.3487839

[1] Our evaluation is done entirely passively through natural experiments, i.e., we do *not* interfere, in any manner whatsoever, in the latency or other aspect of the service experienced by users. We do *not* look at the content of user actions (e.g., email content) or identify users or even analyze individual users. All of our analysis is performed at the level of large user aggregates. For these reasons, we believe our work does not raise any ethical concerns.

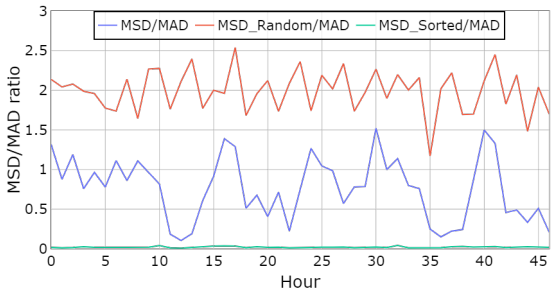

Figure 1: The MSD/MAD ratio of the actual time series of latency seen compared to two extremes: the series shuffled randomly and the series sorted by latency.

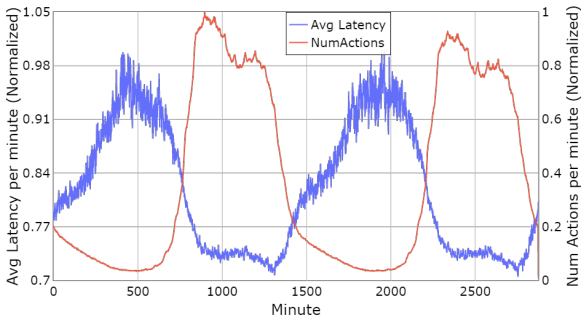

Figure 2: The variation in latency and the rate of user activity over a 2-day period. (The y-axis scale is normalized because of the commercial sensitivity of the latency and user activity data.)

web browser) as the time interval from when a user initiates an action, e.g., clicking a mail item, until the end of the action, e.g., when the mail item is rendered. M refers to a set of minimal, and optional, metadata about the user performing the action, e.g., their subscription type (i.e., whether they are a business user or a consumer user). M helps segregate the analysis based on user groups; in the absence of M, the analysis would be performed for the overall population, without any segregation. Such telemetry is available almost universally in the context of online services, which facilitates the broad application of `AutoSens`.

Central to the working of `AutoSens` is the notion of 'latency bias'. If users dislike high latency, they would tend to perform fewer actions when the latency is high compared to when it is low, thereby exhibiting a 'bias' towards lower latency. Note that if the latency were completely unpredictable, e.g., changing randomly from one moment to the next, the latency would not bias the rate of user actions. The reason is that even if the user has a preference for low latency, they would not be able to exercise this preference, since they would not know the latency of an action (e.g., clicking on a link) in advance. On the flip side, although the latency could, in theory, be predictable in complex ways, we argue that, in practice, it would have to be predictable in a simple way for it to be actionable by human users.

Specifically, locality in the occurrence of low latency could make the latency preference, if any, of users actionable. When the service is fast and responsive, users would likely stay on and do more actions. Conversely, if the service is slow and unresponsive, they might prefer to take a break and come back later.

To analyze the locality in occurrence of low latency, we pursue a two-pronged approach. First, we apply the mean successive difference test on the time series of latency samples corresponding to user actions. This test involves computing the ratio of the mean successive difference (MSD) and the mean absolute difference (MAD) [30]. Intuitively, greater the locality in latency (i.e., similarity in latency level close in time), the smaller the ratio. Second, we compute the temporal density of the latency samples, computed over windows of 1-minute duration, and compare it to the average latency in that window. Intuitively, a negative (or positive) correlation would point to the temporal clustering of the low (or high) latency points.

Figure 1 shows that the MSD/MAD ratio is significantly smaller than it is if the data were shuffled randomly. This suggests that there is a high degree of locality in the latency time series, with periods of low latency interspersed with periods of high latency. (Of course, the ratio would be close to zero, if the data were perfectly sorted by latency.) Figure 2 shows that the periods of low latency tend to have a much higher rate of user activity, and vice versa. Together, these findings suggests that the latency samples obtained from actual user activity tend to be concentrated in periods of low latency and as such provide a "biased" view of the underlying latency distribution, an issue we turn to next.

2.2 Key Idea: Mitigating User Bias

The key idea in `AutoSens` is to construct and then compare the distribution of latency of user actions (*biased* PDF, B) and the inherent or underlying latency distribution independent of user actions (*unbiased* PDF, U).

The PDF, B, is constructed based simply on the latency of each user action, as recorded in the logs. This is termed the biased PDF because it reflects the bias, if any, on the part of users to perform more frequent or less frequent actions based on the latency. In some cases, such bias could arise because of explicit user preference, e.g., users might use a service less when the latency is high. In other cases, the latency, being in the users' critical path, could slow them down and thereby result in fewer actions. For instance, a user who is scanning through the new emails that have arrived in their inbox would get slowed down if the action of clicking on and opening each email takes longer. While in our work, we do not distinguish between these cases because both ultimately impact the number of actions that users perform, we do discuss this further in §3.5.

The unbiased PDF, U, needs to be inferred through indirect means, since it corresponds to what the latency would have been at times that are unrelated to when users actually made accesses and therefore we might not have direct latency measurements for. To address this challenge, we use the following approach to approximate U [2]: we draw samples by repeatedly picking points in time uniformly at random and then picking the latency sample that is closest in time to the chosen time. If there are multiple latency samples at the chosen time, we pick one of the samples at random. This procedure is illustrated in Figure 3(a), where the orange dots correspond to actual latency samples (i.e., these are part of the

[2]For ease of exposition, we use the term "unbiased" for U, though our estimation might only provide an approximation of it.

biased distribution B), the blue crosses denote the random times picked in the construction of U, and the orange dots marked with blue circles are the samples that feed into U.

2.3 Inferring Latency Preference

Both B and U are computed as histograms, with a time bin of 10 ms. Figure 3(b) shows the resulting B and U PDFs. The latency preference corresponding to each latency is computed as the ratio B/U of the corresponding probability densities. As shown in Figure 3(c), this ratio is somewhat noisy. Therefore, we use the Savitzky–Golay filter [25], with a window size of 101 and polynomial degree of 3, to smooth the latency preference estimate. Finally, we pick the preference corresponding to a particular latency as the reference and normalize the preference corresponding to the other latency values to obtain the `normalized latency preference`, which we report in all of our analyses. A `normalized latency preference` of x (e.g., 0.8) at a particular level of latency means that all other factors (e.g., the confounders, discussed in Section 2.4 next) being equal, the user is $(1-x)\times 100\%$ (e.g., 20%) less active at this latency compared to the reference latency.

2.4 Mitigating Confounding Factors

While our interest is in identifying the impact of latency on user activity, there are other confounding factors to contend with. We discuss some of these and the steps we take to mitigate their effect.

2.4.1 Time. The time of day or the day of week could have a significant impact on user activity. For instance, users are less likely to be active during the middle of the night than during daytime. Likewise, users might be less (or, depending on the service, more) active during the weekend than during the weekdays.

Such an impact of time on user activity would not have a bearing on `AutoSens` if it were uncorrelated with latency. However, latency is often a strong function of time; for instance, it would be lower during a less busy hour than during a busy hour because of reduced load and congestion. Therefore, we might see *fewer* accesses by users when latency is low, not because users have an aversion to low latency but because of the time confounder noted above, which could lead `AutoSens` astray.

To mitigate the above issue and allow us to pool together data from across hours, we model the time confounder as a time-based activity factor, α, that reflects how active the user is during a particular time of day. For instance, α would likely be high during the daytime and low in the middle of the night. To estimate *alpha*, we start by discretizing time (into 1-hour time slots) and also latency (into bins of 10 ms). Consider two time slots, T_1 and T_2. Let f_1^L represent the fraction of time in slot T_1 when the latency is L and c_1^L represent the count of user actions with latency L. f_2^L and c_2^L are defined likewise for slot T_2. f_T^L is estimated based on the unbiased latency distribution U_T for that time slot T, e.g., if $U_T(L_a) = 2\times U_T(L_b)$, then latency L_a is said to occur twice as often as L_b, i.e., $f_T^{L_a} = 2\times f_T^{L_b}$.

The temporal rate of user actions corresponding to latency bin L during time slot T_1 is c_1^L/f_1^L, and likewise is c_2^L/f_2^L for time slot T_2. If we treat T_1 as the reference time slot, then we estimate $\alpha_{T_2,L}$ as $\frac{c_2^L/f_2^L}{c_1^L/f_1^L}$. Intuitively, this is the ratio of the temporal rates of actions for the same latency bin across the two hours. Therefore, the difference in the rates is attributable to the time confounder. We estimate $\alpha_{T_2,L}$ for each latency bucket L and compute the average across all latency buckets to estimate the overall time-based activity factor, α_{T_2}, corresponding to time slot T_2. (See Section 3.6 for the finding that α remains stable across the latency bins.) To normalize the counts in time slot T_2, we divide c_2^L corresponding to each L by α_{T_2}.

Time slot	Latency	# actions	% time with this latency	Normalized # actions
Day	Low	90	30%	90
Day	High	140	70%	140
Night	Low	26	80%	250
Night	High	4	20%	38

Table 1: Simple example to illustrate normalization to mitigate time confounder. The "day" time slot is used as the reference for normalization.

Once such normalization has been performed for all time slots and latency buckets, we then compute the biased (B) and unbiased (U) distribution by pooling together information across all time slots. The normalization helps neutralize the time confounder. For instance, the low count of user actions in the middle of the night would be replaced a higher count commensurate with the greater prevalence of low latency during nighttime.

Illustration of time-based normalization: Table 1 illustrates normalization with a simple example. Here, time is discretized into two equal-length slots ("day" and "night") and latency also into two bins ("low" and "high").

If we had ignored the time confounder, we would have computed the user's level of activity when latency is "low" as (90+24)/(30+80) = 1.04 actions per unit time, and that when the latency is "high" as (140+4)/(70+20) = 1.6 actions per unit time. This would indicate that the user performs more actions when the latency is "high" than when it is "low", which clearly does not accord with intuition.

Instead, if we treat the "day" time slot as the reference and normalize the counts corresponding to the "night" time slot, the time-based factor would be estimated as $\alpha_{Night,Low} = (26/80)/(90/30) = 0.108$ and $\alpha_{Night,High} = (4/20)/(140/70) = 0.100$, so $\alpha_{Night} = (0.108+0.100)/2 = 0.104$, i.e., the average α across the latency bins. Therefore, the normalized count of actions during the night would be $26/0.104 = 250$ and $4/0.104 = 38$, respectively, for the "low" and "high" latency bins. Combining these normalized counts with those from the day, the user's level of activity would be estimated as $(90+250)/(30+80) = 3.09$ actions per unit time when the latency is "low" and as $(140+38)/(70+20) = 1.97$ actions per unit time when the latency is "high". That is, the level of activity would be higher when the latency is "low" compared to when the latency is "high", just as we would expect.

Note that in general, noise in the data would result in somewhat different results depending on the time slot that is picked as the reference. Therefore, in our analysis, we pick multiple references in turn and then average the results.

2.4.2 Content. The content of user actions could also impact the volume of activity. User actions could be of different types, each entailing a different level of user engagement (e.g., clicking on an email, performing a search, etc.). Also, the content returned by the service could determine future actions, e.g., whether the relevant

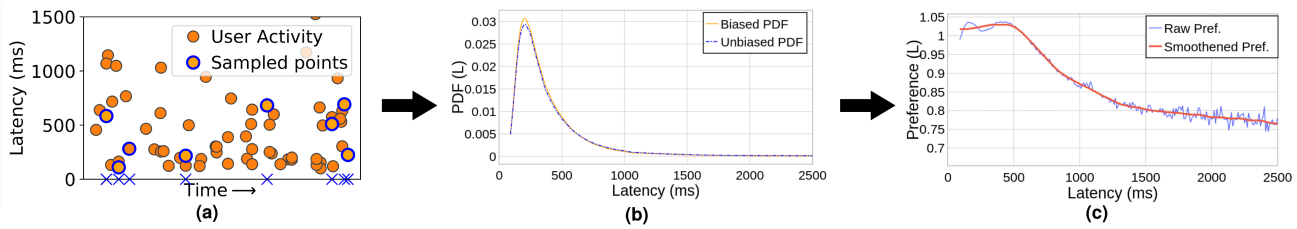

Figure 3: Illustrative figures based on actual data to provide an overview of the AutoSens methodology: (a) Approximating the unbiased distribution by drawing samples (blue circles) at random from the biased samples corresponding to actual user activity, (b) the resulting biased (B) and unbiased (U) PDFs, and (c) the latency preference computed as the ratio B/U and then smoothed.

results are returned at the top (in which case the user might be led away from the search service) or the results are a poor match for what the user was looking for (in which case the user might invoke search again, say after refining their search terms).

To mitigate such confounders, in our analysis, we segregate the actions based on their type and analyze each separately (Section 3.2). Indeed, our analysis uncovers interesting differences in latency sensitivity across the action types. In our work, we did not have access to the content of emails or email folders themselves, but in general, one could mitigate the effects of content-based confounding through appropriate segregation. For instance, we could focus on just the search actions for "head" queries, where the relevant results are likely to be returned in the first go.

2.4.3 User Conditioning. User conditioning could have a bearing on the latency sensitivity of users. For instance, if a user has come to expect low latency (say because they have a strong network connection), they might react more negatively than a user who is accustomed poor latency. To address this confounder, our analysis also segregates users based on the latency they have typically experienced (see Section 3.4).

3 EVALUATION

We describe the data set used in our evaluation and then present the results of various analyses.

3.1 OWA Data Set

Our data comprises server-side logs in OWA, a large web-based email service with millions of users the world over. Each log entry records the time stamp when an action is started, type of access (i.e., user action, see Section 3.2), latency, an anonymized GUID of the user, and the user type (see Section 3.3), among other information. Note that the latency is measured at the client side (from the time an action is initiated until a response is received) and then conveyed to the server, where it is logged. We only focus on successful actions; we ignore the cases where an error was returned.

Our analysis is based on logs of several billion user actions over a 2-month period (January and February, 2021). Due to reasons of commercial sensitivity, we are not in a position to report precise statistics on the usage of the service.

In most of the analyses presented here, we pool in data across different times of day to effectively compute latency preference averaged across these times. However, in Section 3.6, we briefly discuss the variation in latency preference across time of day.

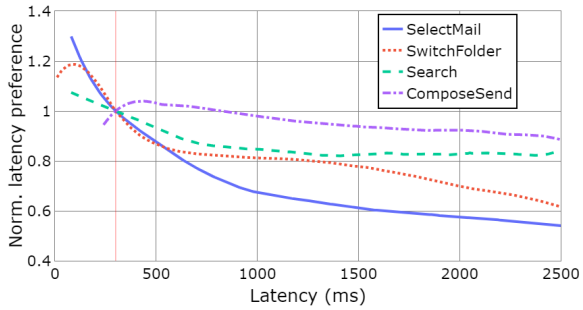

Figure 4: `normalized latency preference` across action types for "business" users in the U.S. during Feb 2021. The reference latency used for normalization is 300 ms.

3.2 Impact of Action Types

Of the various action types supported by OWA, we focus on four: (1) `SelectMail` (click and open email item in the browser), (2) `SwitchFolder` (click and switch folder), (3) `Search` (perform search over mailbox content), and (4) `ComposeSend` (click to send email).

Figure 4 shows the `normalized latency preference` as a function of latency. The normalization is done with respect to a reference latency of 300 ms. We see that `normalized latency preference` drops most sharply for `SelectMail` and then for `SwitchFolder`, perhaps reflecting the expectation of "instantaneous" response users have for these actions. For instance, as the latency grows to 500 ms, 1000 ms, and 1500 ms, respectively, `normalized latency preference` drops to 0.88, 0.68, and 0.61, respectively. This indicates that the increase in latency to 500 ms, 1000 ms, and 1500 ms reduces the incidence of user activity by 12%, 32%, and 39%, respectively, relative to the reference latency of 300 ms.

On the other hand, the `Search` action shows a much less steep drop off, suggesting that users are conditioned to tolerating a somewhat higher latency for the search operation.

Finally, `ComposeSend` is an asynchronous operation, wherein the user interface returns control to the user even as the email is queued up and sent in the background. This may explain why `normalized latency preference` remains nearly flat, indicating that there is little sensitivity to latency.

3.3 Business versus Consumer Users

OWA includes both a commercial service, wherein business users pay for subscription, and a free service aimed at consumers. Figure 5

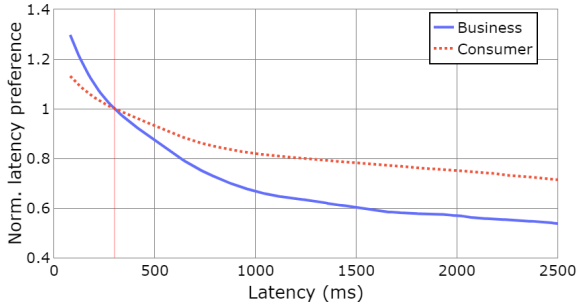

Figure 5: The `normalized latency preference` for the `SelectMail` action by business users and consumers in the U.S. during February 2021.

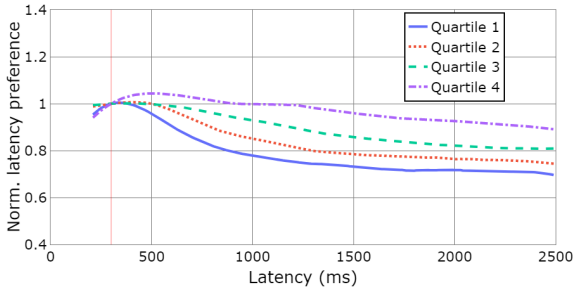

Figure 6: The `normalized latency preference` for groups of U.S. consumer users in February 2021, segregated into quartiles based on their median latency.

graphs the `normalized latency preference`, as a function of latency, for the `SelectMail` action by these two categories of users in the U.S. We see that the drop off is sharper for the business users. Possibly, consumer users are more tolerant of latency because they are obtaining the service for free.

3.4 Conditioning to Speed

Next, we consider the impact of conditioning to speed. That is, how does getting "used to" higher or lower speeds impact the sensitivity of users to latency? To perform this analysis, we first segregate users into quartiles based on their median latency, with Q1 corresponding to the lowest latency (i.e., fastest speed) and Q4 to the highest latency (i.e., slowest speed). We use an anonymized user-identifier to compute the per-user median latency, which enables grouping users into quartiles. As stated before, we do not look into the content of individual user's actions or analyze users individually. We only analyze large aggregates of users.

Figure 6 plots the `normalized latency preference` for users across the latency quartiles. We see a consistent trend, with the sensitivity to latency decreasing progressively as we go from Q1 to Q4. (Note that the comparison is being made for the same x value, i.e., latency; indeed, even Q4 users experience low latency at times.) In other words, users who are used to a lower latency tend to be more sensitive to latency, which accords with intuition.

3.5 Latency Preference vs. Latency Bottleneck

As discussed in §2.2, in this work, we do not make a distinction between the 'true' latency preference of a user impacting the volume

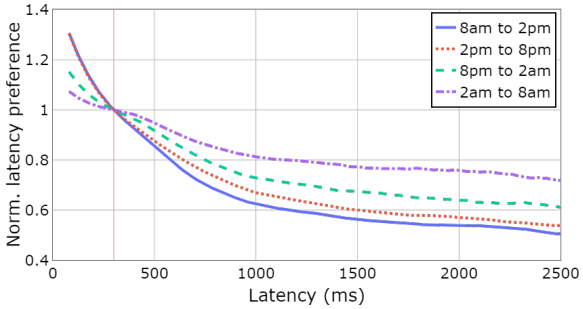

Figure 7: The `normalized latency preference` for the `SelectMail` action by U.S. business users in February 2021, across different times of day.

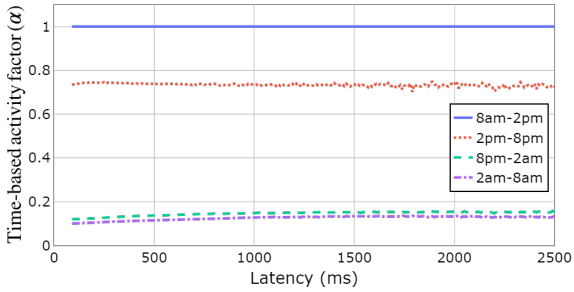

Figure 8: The time-based activity factor, α, for the `SelectMail` action by U.S. business users in February 2021, across different times of day (with the 8am to 2pm period taken as reference).

of user activity versus the user activity being bottlenecked by the latency and hence dropping as the latency increases. Nevertheless, our results do point to the former being a significant factor.

For instance, if we consider the `SelectMail` action in Figure 4, we see that the `normalized latency preference` drops from 0.87 to 0.67 (a factor of 1.3), as the latency (i.e., the duration from the initiation of a user action, such as clicking, until its completion) increases from 500 ms to 1000 ms and then further drops to 0.59 as the latency doubles again to 2000 ms (a further factor of 1.1). If the user were indeed bottlenecked by this end-to-end latency, we would expect the volume of user activity, and hence the `normalized latency preference` as computed, to drop much more sharply — by a factor of two for each doubling of latency.

Furthermore, we see that for the same latency, the `normalized latency preference` is significantly different across action types (Figure 4) and across user groups (Figure 5). This again suggests that actual user latency preference is a significant factor, and it is not just that the user is bottlenecked on the latency.

3.6 Effect of Time of Day

While our analysis thus far has pooled together data from all hours, we now segregate data across different times of day. We consider four 6-hour periods: 8am-2pm, 2pm-8am, 8pm-2am, 2am-8am (all with respect to local time of the user) and examine two questions.

First, Figure 7 shows the normalized latency preference for `SelectMail` for business users across these 4 periods. In each period, we see a consistent trend, with the preference decreasing as

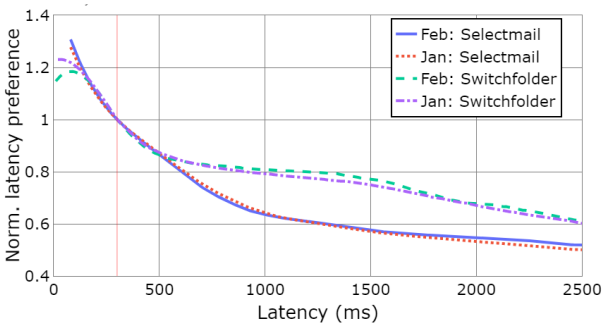

Figure 9: Stability over months

latency increases. However, the drop is sharper during the daytime periods than it is during the nighttime ones. Perhaps users who are active late in the night have a compelling reason to do so and therefore are less sensitive to latency. We also see that the overall latency preference curve obtained by pooling together data from all hours (`SelectMail` curve in Figure 4) lies in between the range of curves in Figure 7, which is as expected.

Second, we compute the time-based activity factor, α (Section 2.4.1), for each period, with the 8am to 2pm period taken as reference. We see from Figure 8 that, as expected, α is lower during the late night periods, reflecting the lower level of user activity then, regardless of the latency. Furthermore, α remains flat across the latency range, lending support to our procedure in Section 2.4.1 of estimating the overall α for a period by averaging across latency bins.

3.7 Consistency across Months

Figure 9 plots the `normalized latency preference` for the `SelectMail` and `SwitchFolder` actions for January and February 2021. We find strong consistency in the drop off in `normalized latency preference` across these months, which suggests that the sensitivity to latency remains stable over the time frame considered.

4 RELATED WORK

Active experiments: Much of past work has been based on active intervention, either by artificially varying network latency or by actively polling user experience. Various authors [3, 9, 10, 20, 28] performed laboratory experiments, while those in [29] conducted surveys, aimed at understanding the relationship between web search latency (which can potentially also affect search quality) and user experience. The studies in [21, 22] performed analogous controlled experiments, while [15] explored crowdsourcing, to model user quality of experience (QoE) for video streaming and its dependence with network performance. In [2, 4], the authors introduced artificial delays in web search and studied the extent to which user experience is impacted.

Passive studies: The problem of inferring user-sensitivity to network performance is well-studied in the specific context of video streaming. For instance, the authors of [26] studied the effect of 31 different network factors on user behavior in mobile video, and also modeled the relationships between such factors and user engagement (specifically video abandonment). Similarly, in [18, 27], the authors apply quasi-experimental designs to investigate causal impacts of stream quality on user playback behavior and account for confounders such as geography, connection type and consumed content. In [11, 12, 23], the authors use logs from streaming views to infer the relationship between system performance and user behavior. Some methods [1, 5, 6] built machine learning models to predict QoE from network quality metrics. In [13], the authors analyzed YouTube user sessions from the point of view of an edge network to guide network capacity planning and design. The work in [24] offered a study of various metrics of user activity in large IPTV systems, in addition to constructing an IPTV user activity workload generation tool. Apart from video, the work in [8] leveraged natural experiments to determine causality from broadband service characteristics to pricing and user demand. Finally, [2, 4, 7] also conducted passive experiments to establish that users are more likely to perform clicks on search result pages served with lower latency, and that this varies with user, query and context.

User-perceived latency metrics: Some attempts have been made at quantifying the users' perception of latency on the web. In [17], the authors propose a user-perceived version of Page Load Time (PLT) by analyzing users' eye gaze as a proxy for their interest, and show that it correlates poorly with traditional PLT metrics. The authors [16] go on to measure PLT for mobile users by proposing a model which incorporates a mobile user's scrolling behavior to view multiple viewports. Both works also demonstrate how such metrics can be used to guide web page optimization.

`AutoSens` in comparison to prior work: `AutoSens` is based purely on natural experiments, so there is no active intervention or modification of the user experience. Furthermore, unlike much of prior work, which has primarily focused on "non-sticky" services like streaming and search (where a user can easily abandon the service when facing very poor network conditions, potentially migrating to a competing service), we have demonstrated our approach on a "sticky" service — email. In such a context, studying user sensitivity is not as simple as monitoring abandonment, e.g., a user facing high latency cannot, at least in the short-term, migrate their entire mailbox to a different service. While we focus on email in this work, we believe our approach can, in principle, be applied to both sticky (e.g., email) and non-sticky services (e.g., web search).

5 CONCLUSION

In this paper, we have presented `AutoSens`, which uses natural experiments to infer the latency sensitivity of users in the context of online services. Specifically, `AutoSens` compares the "biased" latency distribution of user actions to an estimate of the underlying "unbiased" latency distribution, which then enables us to compute the `normalized latency preference` corresponding to a given latency. We apply `AutoSens` in the context of Microsoft `OWA`. The findings from our various analyses accord with intuition.

ACKNOWLEDGEMENTS

We thank our shepherd, Zachary Bischof, and the anonymous IMC reviewers for their insightful comments. We also thank Sreangsu Acharyya, Jim Kleewein, Radhika Garg, Dave Meyers, Jared Mitchell, and Amit Sharma, all from Microsoft, for their helpful inputs.

REFERENCES

[1] V. Aggarwal, E. Halepovic, J. Pang, S. Venkataraman, and H. Yan. Prometheus: Toward quality-of-experience estimation for mobile apps from passive network measurements. In *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, pages 1–6, 2014.

[2] I. Arapakis, X. Bai, and B. B. Cambazoglu. Impact of response latency on user behavior in web search. In *Proceedings of the 37th international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 103–112, 2014.

[3] L. Azzopardi, D. Kelly, and K. Brennan. How query cost affects search behavior. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 23–32, 2013.

[4] X. Bai, I. Arapakis, B. B. Cambazoglu, and A. Freire. Understanding and leveraging the impact of response latency on user behaviour in web search. *ACM Transactions on Information Systems (TOIS)*, 36(2):1–42, 2017.

[5] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang. A quest for an internet video quality-of-experience metric. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pages 97–102, 2012.

[6] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang. Developing a predictive model of quality of experience for internet video. *ACM SIGCOMM Computer Communication Review*, 43(4):339–350, 2013.

[7] M. Barreda-Ángeles, I. Arapakis, X. Bai, B. B. Cambazoglu, and A. Pereda-Baños. Unconscious physiological effects of search latency on users and their click behaviour. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 203–212, 2015.

[8] Z. S. Bischof, F. E. Bustamante, and R. Stanojevic. Need, want, can afford: Broadband markets and the behavior of users. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 73–86, 2014.

[9] J. D. Brutlag, H. Hutchinson, and M. Stone. User preference and search engine latency. 2008.

[10] A. Crescenzi, D. Kelly, and L. Azzopardi. Time pressure and system delays in information search. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 767–770, 2015.

[11] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the impact of video quality on user engagement. *ACM SIGCOMM Computer Communication Review*, 41(4):362–373, 2011.

[12] A. Finamore, M. Mellia, M. M. Munafo, R. Torres, and S. G. Rao. Youtube everywhere: Impact of device and infrastructure synergies on user experience. In *Proceedings of the 2011 Conference on Internet measurement conference*, pages 345–360, 2011.

[13] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. Characterizing user sessions on youtube. In *Multimedia Computing and Networking 2008*, volume 6818, page 681806. International Society for Optics and Photonics, 2008.

[14] Google. Google I/O '08 Keynote by Marissa Mayer. https://youtu.be/6x0cAzQ7PVs?t=936, 2021. [Online; accessed 25-May-2021].

[15] T. Hoßfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz. Quantification of YouTube QoE via crowdsourcing. In *2011 IEEE International Symposium on Multimedia*, pages 494–499. IEEE, 2011.

[16] C. Kelton, J. Ryoo, A. Balasubramanian, X. Bi, and S. R. Das. Modeling user-centered page load time for smartphones. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 1–12, 2020.

[17] C. Kelton, J. Ryoo, A. Balasubramanian, and S. R. Das. Improving user perceived page load times using gaze. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 545–559, 2017.

[18] S. S. Krishnan and R. K. Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *IEEE/ACM Transactions on Networking*, 21(6):2001–2014, 2013.

[19] G. Linden. Geeking with greg. http://glinden.blogspot.com/2006/11/marissa-mayer-at-web-20.html, 2021. [Online; accessed 25-May-2021].

[20] D. Maxwell and L. Azzopardi. Stuck in traffic: How temporal delays affect search behaviour. In *Proceedings of the 5th Information Interaction in Context symposium*, pages 155–164, 2014.

[21] R. K. Mok, E. W. Chan, and R. K. Chang. Measuring the quality of experience of http video streaming. In *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*, pages 485–492. IEEE, 2011.

[22] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang. Inferring the qoe of http video streaming from user-viewing activities. In *Proceedings of the 1st ACM SIGCOMM Workshop on Measurements Up The Stack*, pages 31–36, 2011.

[23] H. Nam, K.-H. Kim, and H. Schulzrinne. Qoe matters more than qos: Why people stop watching cat videos. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2016.

[24] T. Qiu, Z. Ge, S. Lee, J. Wang, J. Xu, and Q. Zhao. Modeling user activities in a large iptv system. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, pages 430–441, 2009.

[25] A. Savitzky and M. Golay. Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36(8):1627–39, 1964.

[26] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang. Understanding the impact of network dynamics on mobile video user engagement. *ACM SIGMETRICS Performance Evaluation Review*, 42(1):367–379, 2014.

[27] R. K. Sitaraman. Network performance: Does it really matter to users and by how much? In *2013 5th International Conference on Communication Systems and Networks (COMSNETS)*, pages 1–10. IEEE, 2013.

[28] N. J. Taylor, A. R. Dennis, and J. W. Cummings. Situation normality and the shape of search: The effects of time delays and information presentation on search behavior. *Journal of the American Society for Information Science and Technology*, 64(5):909–928, 2013.

[29] J. Teevan, K. Collins-Thompson, R. W. White, S. T. Dumais, and Y. Kim. Slow search: Information retrieval without time constraints. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval*, pages 1–10, 2013.

[30] J. von Neumann, R. H. Kent, H. R. Bellinson, and B. I. Hart. The mean square successive difference. *The Annals of Mathematical Statistics*, 12(2):153–162, 1941.

[31] J. Young and T. Barth. Akamai online retail performance report: Milliseconds are critical. https://www.akamai.com/uk/en/about/news/press/2017-press/akamai-releases-spring-2017-state-of-online-retail-performance-report.jsp, 2021. [Online; accessed 25-May-2021].