Detecting Self-Interruptions during Reading

CPSC 539 Project Report

Jan Pilzer
University of British Columbia
Vancouver, Canada
pilzer@cs.ubc.ca

Sam Liu
University of British Columbia
Vancouver, Canada
xinhliu@cs.ubc.ca

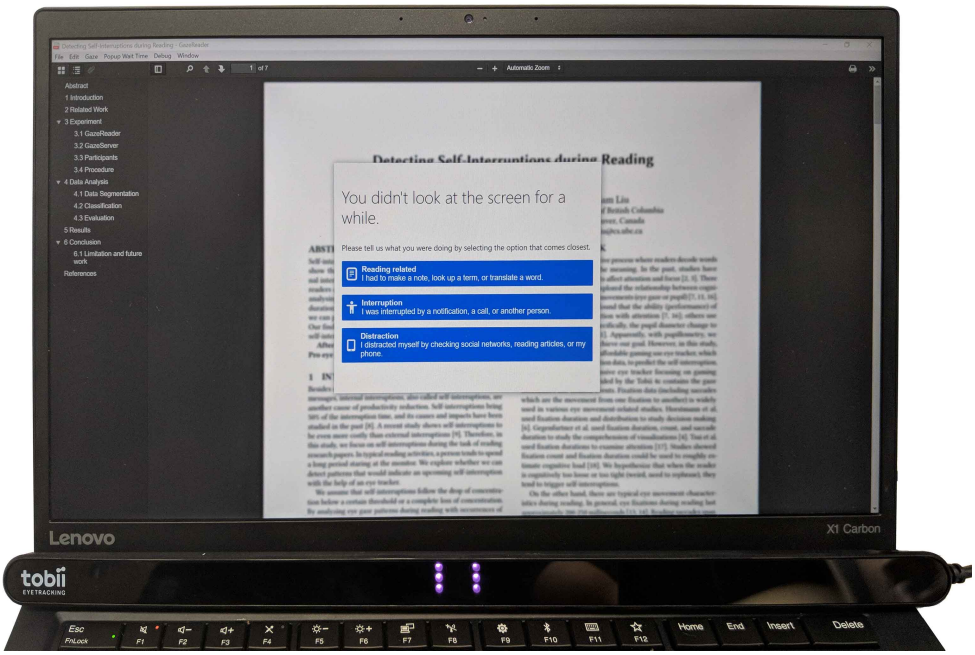

Figure 1: *GazeReader* setup using the Tobii Eye Tracker 4C after the participant didn't look at the screen for some time.

ABSTRACT

Self-interruptions bring 50% of the interruption time, and studies show self-interruptions to be even more costly than external interruptions. Reading is a complex cognitive process where readers decode words and symbols and derive the meaning. By analysing the eye movement, specifically the fixation duration and saccades duration, length, and angles we demonstrate that we can potentially detect the self-interruption before they happen. Our findings could be used to help developers predict the coming self-interruptions and help them arrange tasks and rests. The precision could be improved by collecting more data and adding additional features such as the pupil dilation. Those however are only available with more advanced eye trackers for now.

1 INTRODUCTION

Besides external interruptions from colleagues, phone calls, and messages, internal interruptions, also called self-interruptions, are another cause of productivity reduction. Self-interruptions bring 50% of the interruption time, and its causes and impacts have been studied in the past [9]. A recent study shows self-interruptions to be even more costly than external interruptions [10]. Therefore, in this study, we focus on self-interruptions during the task of reading research papers. In typical reading activities, a person tends to spend an extended period staring at the monitor. We explore whether we can detect patterns that would indicate an upcoming self-interruption with the help of an eye tracker.

We assume that self-interruptions follow the drop of concentration below a certain threshold or a complete loss of concentration. By analyzing eye gaze patterns during reading with occurrences of self-interruptions we hope to find some specific movement patterns that correlate to the concentration loss. Below are the research questions we expect to find the answers for through our study.

Research Questions

- *(RQ1)* Does the low-cost eye tracker have the sufficient accuracy to track the current reading line?
- *(RQ2)* Does eye gaze behave differently before a self-interruption?
- *(RQ3)* Can we detect eye gaze patterns that occur before a self-interruption?

The rest of the paper is organized as follows. First, we discuss the related work. Then we illustrate our experiment design, data preparation and data analysis in detail. Afterwards, our results

demonstrate the feasibility to detect a self-interruption by the eye movement data before it. Finally, we discuss the limitations and potential future directions.

2 RELATED WORK

Reading is a complex cognitive process where readers decode words and symbols and derive the meaning. In the past, studies have shown that cognitive aspects affect attention and focus [2, 3]. There are also researchers who explored the relationship between cognitive characteristics and eye movements (eye gaze or pupil) [8, 13, 18]. For instance, researchers found that the ability (performance) of eye-tracking has a connection with attention [8, 18]; others use cognitive pupillometry, specifically, the pupil diameter change to estimate cognitive load [13]. Apparently, with pupillometry, we should be more likely to achieve our goal. However, in this study, we explore the usage of the affordable gaming use eye tracker, which only provides eye gaze position data, to predict the self-interruption. The Tobii 4c is an inexpensive eye tracker focusing on the gaming experience. The data provided by the Tobii 4c contains the gaze coordinates and fixation events. Fixation data (including saccades which are the movement from one fixation to another) is widely used in various eye movement-related studies. Horstmann et al. used fixation duration and distribution to study decision making [7]. Gegenfurtner et al. used fixation duration, count, and saccade duration to study the comprehension of visualizations [4]. Tsai et al. used fixation durations to examine attention [19]. Studies showed fixation count and fixation duration could be used to roughly estimate cognitive load [20]. We hypothesize that when the reader is cognitively too loose or too tight (weird, need to rephrase), they tend to trigger self-interruptions.

On the other hand, there are typical eye movement characteristics during reading. In general, eye fixations during reading last approximately 200-250 milliseconds [15, 16]. Reading saccades span on average about a span of 7 to 9 letter spaces [12]. Besides, most of the saccades are from left to right (top to bottom if there is a line break), except 10-15% regression [16]. This feature could also be used to detect the abnormality potentially. If the fixation frequency, duration, and saccade direction, span become abnormal when cognitive condition changes, they can help to detect self-interruptions. Essentially, any eye movement abnormality could potentially lead to the loss of concentration, which is likely to induce self-interruption further. In our study, all these "typical" characteristics mentioned above will be used for analysis.

3 EXPERIMENT

To facilitate the tracking of reading patterns, we develop a program to record the coordinates of the eye gaze and link them to the content of the paper at that point that can run on both Windows and macOS: *GazeReader*. To detect self-interruptions, *GazeReader* records the title of the active window and the times of missing eye gaze. We use window switches (to unrelated apps/websites) as a sign of self-interruptions. Those switches could be to other work-related tasks (writing an email) or recreational (checking a social network), but we are not differentiating between those cases at this point. A break in the reading during which the eye gaze is not on the screen could imply a self-interruption (checking the

Figure 2: Tobii Eye Tracker 4C

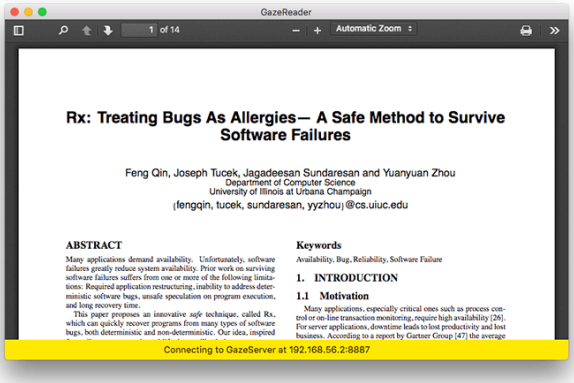

Figure 3: *GazeReader* interface

smartphone) or thinking about the content of the reading. The distinction of those cases requires a manual review of the recording by the participant. Here we use a pop-up window to ask participants for details to determine if it is a self-interruption or not. The technical details will be illustrated in subsection 3.1. Apart from *GazeReader*, a standalone server is responsible for communication with the Tobii Eye Tracker 4C and redirection of the data stream to the *GazeReader*. The technical details will be illustrated in subsection 3.2. We recruit 7 participants including ourselves to read total 19 papers, and each paper is finished in one reading session. Participants borrow a Tobii 4c (see Figure 2[1]) to attach to the bottom bezel of their monitors. The details of participants and procedure will be illustrated in subsection 3.3 and subsection 3.4.

3.1 GazeReader

In order to let more people participate our study, we decide to use *electron*[2] to build our *GazeReader* as it allows for easy operating system independent creation of distributable applications. The whole *GazeReader* is written in JavaScript and we use *PDF.js*[3] to display PDF and extract the text layers.

Figure 3 shows the basic interface of *GazeReader*. During the reading, users might zoom, scroll, resize the window, or switch to other windows. These events are tracked and recorded accordingly:

Blur. This event reports when the *GazeReader* becomes inactive (when the user focuses on the other window).

[1] https://tobiigaming.com/eye-tracker-4c
[2] https://electronjs.org
[3] https://github.com/mozilla/pdf.js

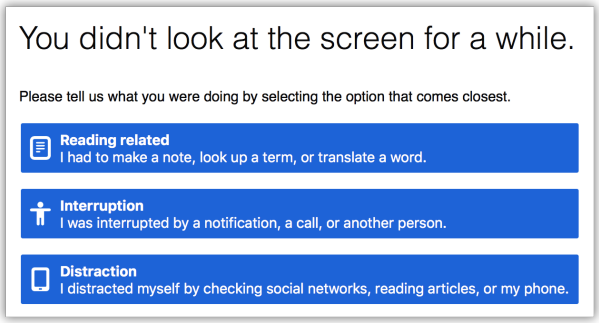

Figure 4: *GazeReader* makes a pop-up window to ask why the participant looks away

Focus. This event reports when the *GazeReader* becomes the active window.

Active. This event reports the currently active window ID and the window title.

Gaze. This event occupies the most of the data, reporting the position of the gaze (both relative and absolute value) and the text layer underneath.

Fixation start. This event reports a fixation start with the same parameters as a gaze event.

Fixation change. This event reports a fixation data with the same parameters as a gaze event.

Fixation end. This event reports a fixation end with the same parameters as a gaze event.

Head pose. This event reports the current head position and rotation.

Scroll. This event reports the previous and current vertical scroll value in pixel and percentage.

Zoom. This event reports the previous and current zoom selection. This could be a zoom factor or a setting like "page fit".

PDF size/position. This event reports a PDF size or position change, and it is triggered by scrolling, window resizing or zooming.

Absence Reason. This event reports the user-selection of a reason for their absence.

3.2 GazeServer

GazeServer is a program to communicate with Tobii 4c and get the necessary data. The reason to make it a standalone program is because the Tobii SDK[4] is only available for Windows. In order to deploy our *GazeReader* to the Mac, we separate the communication component that talks to Tobii 4c to *GazeServer* which runs on a Windows virtual machine hosted on the Mac. For Windows user however, they would not sense the existence of *GazeServer*. The *GazeReader* and *GazeServer* communicate using WebSocket. The *GazeServer* is written in *C* with *.NET* framework.

3.3 Participants

5 male and 2 female students participant in this study. 2 current are undergraduate student and the other 5 are graduate student. All participants are in their twentieth.

3.4 Procedure

Each participant is asked to sign the consent form before beginning the experiment. We help the participant setting up the required software on their machine. This includes the eye tracking software and the *GazeReader* in case of Windows or a Windows virtual machine, which contains the eye tracking software and the *GazeServer*, and the *GazeReader* in case of macOS. Participants are briefly told about the scope of our experiment and shown the selection options of the pop-up (see Figure 4). A participant starts a reading session by running the *GazeReader* and opening a PDF file. After reading one or multiple papers, the session files are passed to us. We have collected over 20 hours reading data (5 million event records) including 54 identified self-interruptions in total.

4 DATA ANALYSIS

4.1 Data Segmentation

Listing 1: A snippet of the session data where <TEXT_LINE> is a line in the paper

```
2017-11-12T01:06:21.913Z|FIXATIONDATA|369.73,715.79;17.47%,8.83%;<TEXT_LINE>
2017-11-12T01:06:21.915Z|FIXATIONEND|332.62,721.53;11.03%,35.74%;<TEXT_LINE>
2017-11-12T01:06:21.915Z|HEAD|6.08,107.60,702.73;-0.27,0.19,-0.07
2017-11-12T01:06:21.918Z|GAZE|357.64,718.33;15.37%,20.74%;<TEXT_LINE>
2017-11-12T01:06:21.933Z|GAZE|326.13,723.11;9.91%,43.14%;<TEXT_LINE>
2017-11-12T01:06:21.938Z|HEAD|6.08,107.60,702.73;-0.27,0.19,-0.07
2017-11-12T01:06:21.986Z|HEAD|6.08,107.60,702.73;-0.27,0.19,-0.07
2017-11-12T01:06:32.174Z|BLUR|
2017-11-12T01:06:32.175Z|ACTIVE|GazeReader.exe;Dialog
2017-11-12T01:37:11.421Z|REASON|distraction
2017-11-12T01:37:11.440Z|FOCUS|
2017-11-12T01:37:11.449Z|GAZE|872.82,534.01;4.50%,7.96%<TEXT_LINE>
2017-11-12T01:37:11.453Z|GAZE|871.96,532.24;2.08%,-0.34%<TEXT_LINE>
2017-11-12T01:37:11.456Z|GAZE|871.53,528.94;0.06%,97.28%;<TEXT_LINE>
2017-11-12T01:37:11.458Z|FIXATIONDATA|871.52,532.08;0.85%,-1.09%<TEXT_LINE>
2017-11-12T01:37:11.462Z|FIXATIONDATA|871.00,529.17;-0.04%,98.36%;<TEXT_LINE>
2017-11-12T01:37:11.467Z|FIXATIONDATA|871.55,524.30;0.06%,75.53%;<TEXT_LINE>
```

Terminologies. The session data created by the *GazeReader* is an event stream. Listing 1 shows a snippet of the session data. To analyze the data, we perform a segmentation with each segment of max length t_i seconds and label each data segment **1** for a segment occurring right before a self-interruption or **0** otherwise. Figure 5 and Figure 6 illustrate how the segmentation is done for the two different types of interruption scenarios.

- **Time of interest** t_i denotes the length of the period before the identified self-interruptions that we are interested in. We label data segments within these periods with **1**. We try $t_i = \{5, 10, 20, 30, 40, 50\}$ to explore how long is optimal for detection.
- **Task switching lag** t_p denotes the interruption lag, which is the time the participant needs to decide to switch to a reading-related tasks. We set $t_i = t_p$. This time is excluded from our analysis. For cases of external interruptions this value does not apply.

[4]http://developer.tobii.com/tobii-sdk-guide/

- **Resumption lag** t_r denotes the resumption lag, t_{r1} for unrelated interruptions and t_{r2} for reading related task switches respectively. When caused by an interruptions, t_r increases as the time of interruption increases, until reach a flat value: around 2 seconds [6, 11]. We allocate extra safety zone by extending t_{r1} to 3 seconds. We choose $t_{r2} = t_{r1}$ for simplicity. So $t_{r1} = t_{r2} = t_r = 3s$.
- **Pop-up threshold** t_w denotes the time during which no gaze is recorded before the pop-up is opened. This is set to ten seconds in the *GazeReader*.
- **Period of interest** is the period before the identified self-interruptions. Each period of interest lengths t_i.
- **Segment 1** is a data segment located within the period of interest. The segment length is equal to t_i.
- **Segment 0** is a data segment within the period we believe the participant is concentrating on reading. A segment 0 lengths t_i.
- The **stripped period** is the period of time we remove from the analysis, including the time when the participant is looking away or resuming from other tasks. To eliminate the effect of task switching lag, during which the participant is less likely to fully concentrate, we also strip that period of data.

In Figure 5, the participant switches from *GazeReader* to another window, so *GazeReader* records a BLUR event. Afterwards, the participant switches back to *GazeReader*, which triggers the FOCUS event. Between BLUR and FOCUS, *GazeReader* regularly records the current active window's title and program ID, which is used to identify whether the participant self-interrupts or not. When we identify the participant self-interrupted, the data within t_i seconds before BLUR event are labeled as **1**. If the participant is doing reading-related tasks such as taking notes, we strip a period of length t_p to remove the effect of task switching lag.

In Figure 6, the participant looks away from the monitor. *GazeReader* waits for t_w until treating the participant as "away." Then *GazeReader* opens a pop-up window (Figure 4) and waits for the participant to select the reason for their absence. As shown in Figure 4, we offer three options to choose from: (1) Reading related (2) Interruption for all external interruptions (3) Distraction for all self-interruptions. If the participant chooses self-interruption, we label the period t_i before the participant's last recorded gaze **0**. If the participant classified the absence as reading-related, we strip a period of length t_p to remove the effect of task switching lag. In all three situations, we strip a period of length t_r due to resumption lag.

All the periods other than those labeled **1** or stripped periods are labeled **0**. Depending on the t_i, we divide each data period into segments of length t_i for classification.

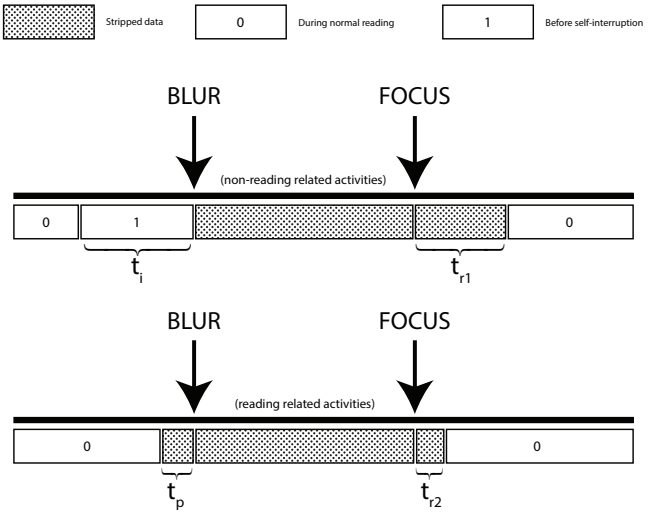

Figure 5: Data segmentation scenario: Switch to another window which is reading related (top) or self interruption (bottom).

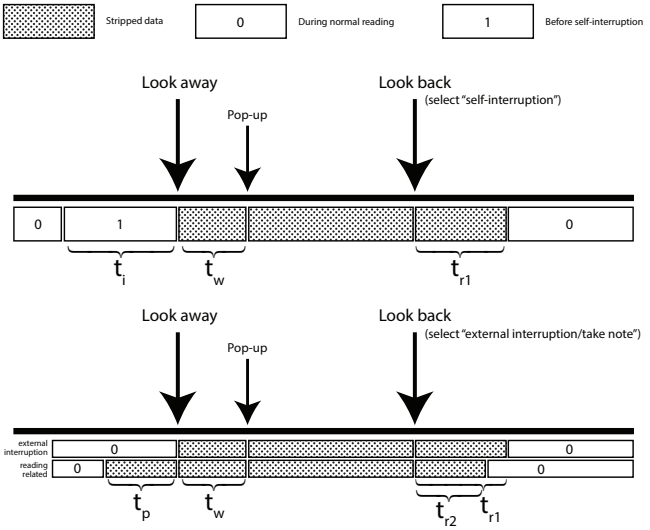

Figure 6: Data segmentation scenario: Looking away from the screen due to self interruption (top), or external interruption or reading related tasks (bottom).

4.2 Classification

Figure 7 illustrates the process of the classification. 2/3 of the data segments are used for training and the remaining 1/3 is used for testing. The process of picking training and testing data are fully random and we repeat this process for several times to examine the stability. In each iteration, the classifier is trained with 2/3 of the data then predict the labels for the 1/3 testing data. The predicted labels are compared with correct labels to calculate **accuracy**. Over several iterations, for each classifier the mean **accuracy** and its standard deviation is calculated. Table 1 shows a summary of the features we are looking at. 10 different classifiers and logistic regression are used for comparison.

(1) AdaBoostClassifier
(2) DecisionTreeClassifier
(3) GaussianNB
(4) GaussianProcessClassifier

Table 1: Features

Features		Sub features				
fixation	duration	mean	median	variance	min	max
	count					
saccade	duration	mean	median	variance	min	max
	length	mean	median	variance	min	max
	angle	mean	median	variance	min	max

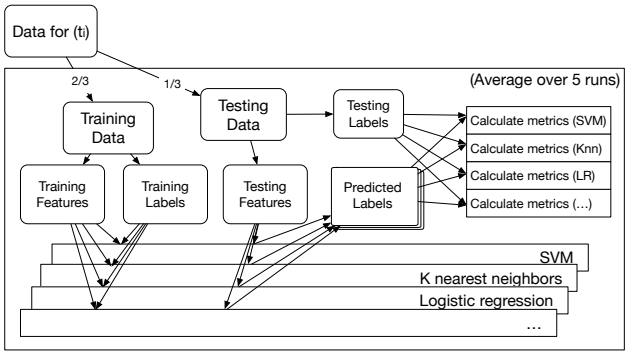

Figure 7: Process of classifications for a (t_i)

(5) KNeighborsClassifier
(6) LogisticRegression
(7) MLPClassifier
(8) QuadraticDiscriminantAnalysis
(9) RandomForestClassifier
(10) SVC (non-linear)
(11) SVC (linear)

The detail configurations for each classifiers are listed in Appendix A. Our work does not focus on machine learning for now, so we do not try to fine tune the parameters further. All machine learning is done using SKLearn [14].

4.3 Evaluation

To evaluate the classifiers, metrics including accuracy, **precision**, **recall** and f1 scores are calculated. **Precision** indicates how many identified self-interruptions are true self-interruptions. **Recall** indicates how many self-interruptions are identified. F1 score is the harmonic mean of **precision** and **recall**: $f_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$. We comparing the metrics when $t_i = 5, 10, 20, 30, 40, 50, 60$ and the results can be found in section 5. In this scenario, we want to have a high **precision** more than **accuracy** and **recall**, because a bad **precision**, which means false identification is common, results in interrupting the readers.

5 RESULTS

GaussianProcessClassier and *non-linear SVC* are removed from all the results because they have abnormal behaviors during the training processes.

Figure 8 shows the **accuracy** curve when t_i changes. At $t_i = 5$, most classifiers except *LogisticRegression* has almost 100% accuracy.

Table 2: Detail scores of classifier AdaBoost, DecisionTree, RandomForest and linear SVC when $t_i = 5$.

Classifier	Accuracy		Precision		Recall		F1	
	mean	std	mean	std	mean	std	mean	std
AdaBoost	1.00	0.00	0.98	0.03	0.97	0.04	0.97	0.02
DecisionTree	1.00	0.00	0.98	0.03	0.97	0.04	0.97	0.02
RandomForest	1.00	0.00	0.92	0.13	0.78	0.17	0.84	0.13
Linear SVC	1.00	0.00	0.82	0.18	0.92	0.09	0.87	0.14

The **accuracy** does not changes noticeably when t_i increases for most classifiers, except for *linear SVC*, *DecisionTree*, and *GaussianNB*, which drops dramatically when $t_i > 5$ ($t_i = 40$ for *GaussianNB*). For **precision** shown in Figure 9, all classifiers have poor **precision** when $t_i > 5$. *AdaBoost*, *DecisionTree*, *RandomForest* and *linear SVC* have the best **precision** (> 80%) among all classifiers. *GaussianNB* has 50% precision, which is equal to guessing. Recall scores in Figure 10 do not reveal a common trend. *Linear SVC* maintains a high **recall** (90%). *DecisionTree*, *GaussianNB*, *LogisticRegression*, *AdaBoost* and *RandomForest* have around 90% **recall** at $t_i = 5$, then drop to lower values when $t_i > 5$. **F1** is a synthetic evaluation combining **precision** and **recall**. Our result (Figure 11) shows a similar **f1** behavior compared to precision: *AdaBoost*, *DecisionTree*, *RandomForest* and *linear SVC* have decent **f1** at $t_i = 5$. For $t_i > 5$, **f1** scores for all classifiers drops to less than 30%.

Overall, there are several classifiers able to perform great and they are *AdaBoost*, *DecisionTree*, *RandomForest* and *linear SVC*.

6 CONCLUSION

In this study, we prove that before a self-interruption, the eye movement is different compared to when the reader concentrates on reading. We demonstrate with proper classifiers, we can detect the incoming self-interruptions by gaining 5-second eye movement data before the interruption event. This would help to investigate the feasibility of predicting self-interruptions. Below we will discuss the limitations and future work.

The recorded data is only relevant if a sufficient number of self-interruptions occurred. When recording very concentrated participants those might be rare. Additionally, knowing about the setup of the experiments might influence the participants to reduce the number of actual interruptions even further. During the reading session, we cannot guarantee that participants are used to reading papers on the screen. According to our oral feedbacks, many participants usually read the print paper. Asking them to read on the screen might make the reading uncomfortable. Furthermore, the Tobii 4c driver only runs on Windows. Near half of our participants use a MacBook and therefore a special virtual machine setup is required to let *GazeReader* function normally. Keeping a VM running is not pleasant, and it might discourage participants who use MacBooks.

During the testing, we found that the Tobii 4c is not accurate enough to pinpoint the exact sentence consistently and 2-3 lines shift are common. The eye tracker can only work precisely at the calibrated distance. When participants change their posture during reading, the accuracy is further reduced. At this point, the accuracy is not sufficient to detect detailed reading patterns in general. But in this study, we focus on cost-effective eye trackers. In the future, we could attempt to use products from the Tobii Pro product line.

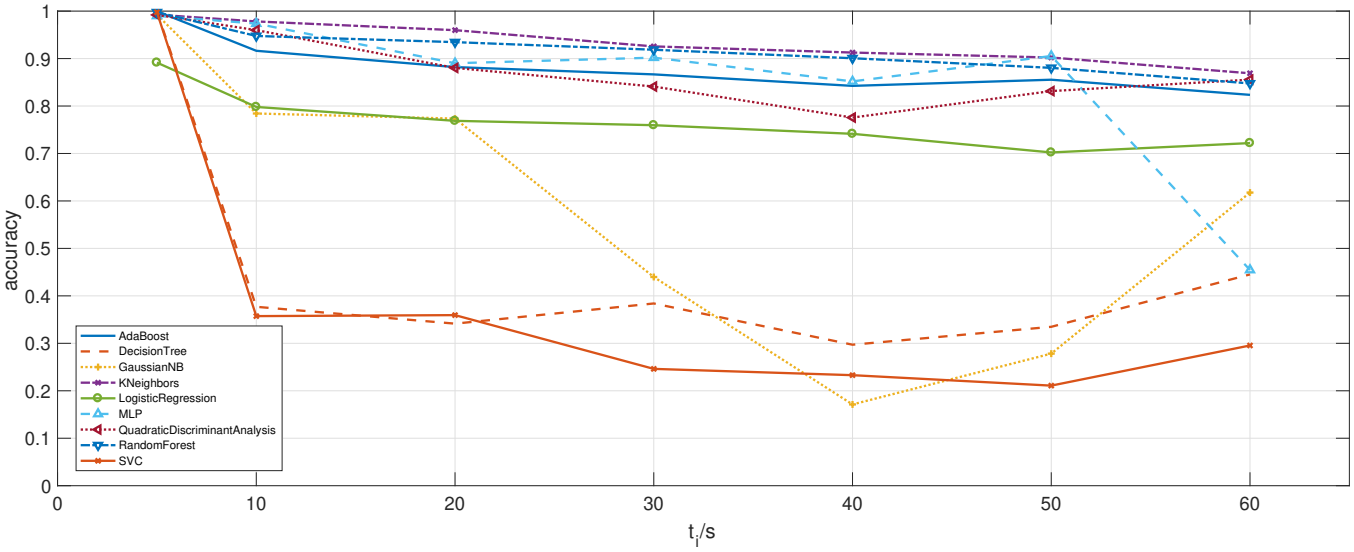

Figure 8: Accuracy scores of different t_is and classifiers. Each line represents the result of a classifier.

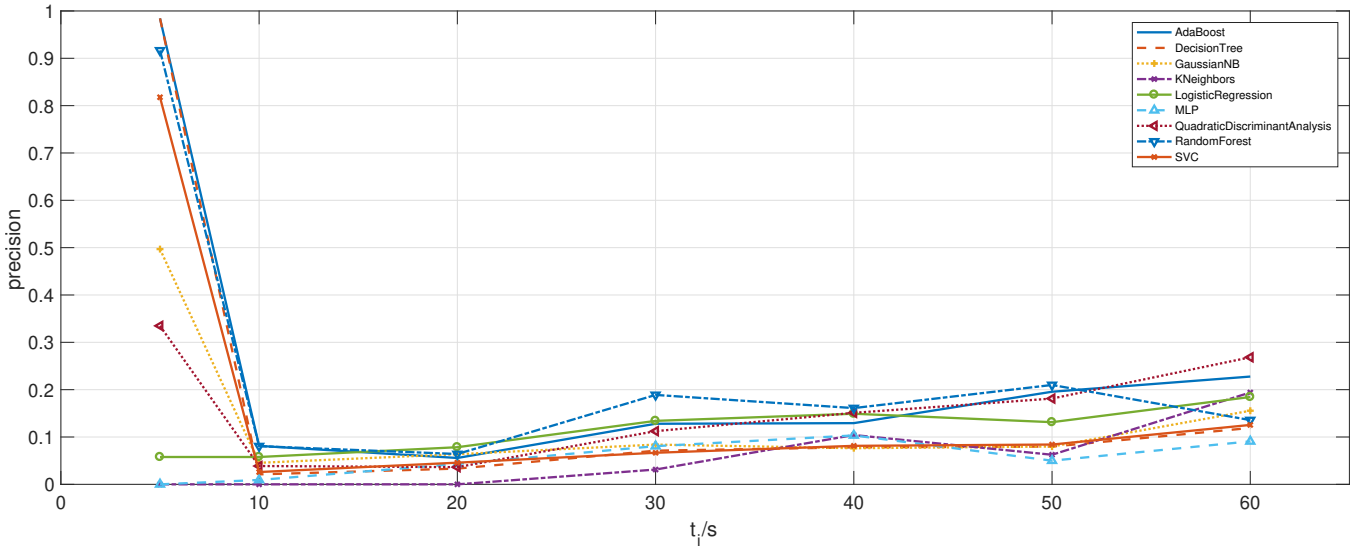

Figure 9: Precision scores of different t_is and classifiers. Each line represents the result of a classifier.

In the data analysis, we use binary classification to detect self-interruptions. However, we believe there are intermediate states. Future work could add more states such as "tend to self-interrupt but fail," which require gathering more information from participants. Besides, the parameters of classifiers are the common values and we did not fine-tune them. The model could be optimized as well. Although the classifiers can achieve decent scores, we do not look into which features play the import roles. We believe a small subset of features we used in this study (which are 17 dimensions) could achieve similar scores but require less training time.

Can eye movement be used to reliably predict self-interruptions and how long in advance can self-interruptions be predicted? In this study, we only study one variable t_i. To answer this question, the future work needs to add different offsets between the period of interest and self-interruption events and compare the classification metrics.

Pupil dilation is typically used as a measure to gauge an individual's interest or arousal in the content they are viewing [5]. In our context, pupil dilation should be a better feature to include. Due to limitations of the eye tracker hardware, we were not able to capture the pupil dilation using the current experimental setup.

Additionally, previous research shows that the eye fixation has different distributions on content words and functional words where functional words are fixated less frequently than content words [1, 17]. A change in the distribution, such as no difference between

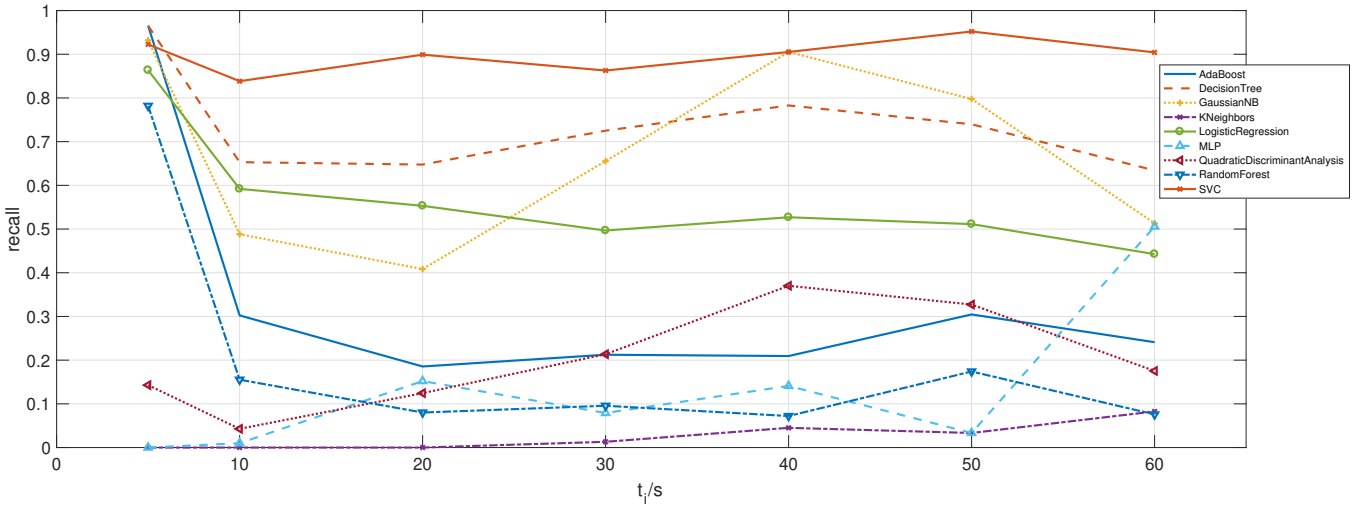

Figure 10: Recall scores of different t_is and classifiers. Each line represents the result of a classifier.

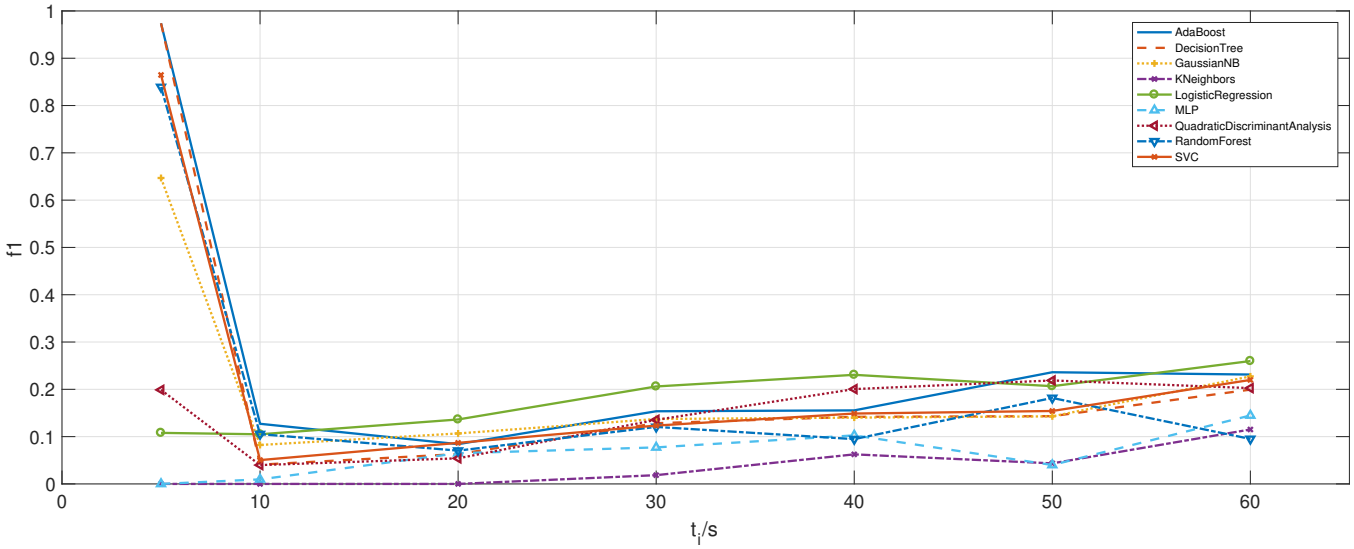

Figure 11: F1 scores of different t_is and classifiers. Each line represents the result of a classifier.

fixation duration, could also be used as a feature in detecting upcoming self-interruptions.

REFERENCES

[1] P.A. Carpenter and M. A. Just. 1983. What your eyes do while your mind is reading. In *Eye movements in reading: Perceptual and language processes*, Keith Rayner (Ed.). Academic Press, New York, 275–307.

[2] James Allan Cheyne, Jonathan S.A. Carriere, and Daniel Smilek. 2006. Absent-mindedness: Lapses of conscious awareness and everyday cognitive failures. *Consciousness and Cognition* 15, 3 (Sept. 2006), 578–592. https://doi.org/10.1016/j.concog.2005.11.009

[3] James W. Danaher. 1980. Human Error in ATC System Operations. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 22, 5 (Oct. 1980), 535–545. https://doi.org/10.1177/001872088002200503

[4] Andreas Gegenfurtner, Erno Lehtinen, and Roger Säljö. 2011. Expertise Differences in the Comprehension of Visualizations: a Meta-Analysis of Eye-Tracking Research in Professional Domains. *Educational Psychology Review* 23, 4 (Dec. 2011), 523–552. https://doi.org/10.1007/s10648-011-9174-7

[5] Laura A. Granka, Thorsten Joachims, and Geri Gay. 2004. Eye-tracking analysis of user behavior in WWW search. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 478–479.

[6] Helen M. Hodgetts and Dylan M. Jones. 2003. Interruptions in the Tower of London Task: Can Preparation Minimise Disruption? *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 47, 8 (Oct. 2003), 1000–1004. https://doi.org/10.1177/154193120304700810

[7] Nina Horstmann, Andrea Ahlgrimm, and Andreas Glöckner. 2009. How distinct are intuition and deliberation? An eye-tracking analysis of instruction-induced decision modes. *An eye-tracking analysis of instruction-induced decision modes* (2009).

[8] William G. Iacono and David T. Lykken. 1979. Eye tracking and psychopathology: New procedures applied to a sample of normal monozygotic twins. *Archives of General Psychiatry* 36, 12 (1979), 1361–1369.

[9] Jing Jin and Laura A. Dabbish. 2009. Self-interruption on the computer: a typology of discretionary task interleaving. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 1799–1808.

[10] Ioanna Katidioti, Jelmer P. Borst, Marieke K. van Vugt, and Niels A. Taatgen. 2016. Interrupt me: External interruptions are less disruptive than self-interruptions. *Computers in Human Behavior* 63 (Oct. 2016), 906–915. https://doi.org/10.1016/j.chb.2016.06.037

[11] Christopher A. Monk, J. Gregory Trafton, and Deborah A. Boehm-Davis. 2008. The effect of interruption duration and demand on resuming suspended goals. *Journal of Experimental Psychology: Applied* 14, 4 (2008), 299–313. https://doi.org/10.1037/a0014402

[12] Robert E. Morrison and Keith Rayner. 1981. Saccade size in reading depends upon character spaces and not visual angle. *Perception & Psychophysics* 30, 4 (July 1981), 395–396. https://doi.org/10.3758/BF03206156

[13] Oskar Palinko, Andrew L. Kun, Alexander Shyrokov, and Peter Heeman. 2010. Estimating cognitive load using remote eye tracking in a driving simulator. In *Proceedings of the 2010 symposium on eye-tracking research & applications*. ACM, 141–144.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[15] Alexander Pollatsek, Keith Rayner, and William E. Collins. 1984. Integrating pictorial information across eye movements. *Journal of Experimental Psychology: General* 113, 3 (1984), 426–442. https://doi.org/10.1037/0096-3445.113.3.426

[16] Keith Rayner. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin* 124, 3 (1998), 372–422. https://doi.org/10.1037/0033-2909.124.3.372

[17] Keith Rayner and S. A. Duffy. 1988. On-line comprehension processes and eye movements in reading. In *Eye movements in reading: Perceptual and language processes*. Academic Press, New York, 13–66.

[18] Charles Shagass, Richard A. Roemer, and Marco Amadeo. 1976. Eye-tracking performance and engagement of attention. *Archives of General Psychiatry* 33, 1 (1976), 121–125.

[19] Meng-Jung Tsai, Huei-Tse Hou, Meng-Lung Lai, Wan-Yi Liu, and Fang-Ying Yang. 2012. Visual attention for solving multiple-choice science problem: An eye-tracking analysis. *Computers & Education* 58, 1 (Jan. 2012), 375–385. https://doi.org/10.1016/j.compedu.2011.07.012

[20] Qiuzhen Wang, Sa Yang, Manlu Liu, Zike Cao, and Qingguo Ma. 2014. An eye-tracking study of website complexity from cognitive load perspective. *Decision Support Systems* 62 (June 2014), 1–10. https://doi.org/10.1016/j.dss.2014.02.007

A CLASSIFIERS

Table 3: Classifiers used in the classification and their configurations

Classifier	Parameters
AdaBoostClassifier	*default*
DecisionTreeClassifier	`max_depth=5, class_weight="balanced"`
GaussianNB	*default*
GaussianProcessClassifier	*default*
KNeighborsClassifier	`n_neighbors=3`
LogisticRegression	*default*
MLPClassifier	`alpha=1`
QuadraticDiscriminantAnalysis	*default*
RandomForestClassifier	`max_depth=5`
SVC (non-linear)	`gamma=2, class_weight="balanced"`
SVC (linear)	`C=0.025, kernel="linear", class_weight="balanced"`