IntelliEye

Enhancing MOOC Learners' Video Watching Experience with Real-Time Attention Tracking

Robal, Tarmo; Zhao, Yue; Lofi, Christoph; Hauff, Claudia

DOI
10.1145/3209542.3209547

Publication date
2018

Document Version
Submitted manuscript

Published in
HT'18

Towards Real-time Webcam-based Attention Tracking in Online Learning

Tarmo Robal
Tallinn University of Technology
Tallinn, Estonia
tarmo@ati.ttu.ee

Yue Zhao
WIS, TU Delft
Delft, Netherlands
y.zhao-1@tudelft.nl

Christoph Lofi
WIS, TU Delft
Delft, Netherlands
c.lofi@tudelft.nl

Claudia Hauff
WIS, TU Delft
Delft, Netherlands
c.hauff@tudelft.nl

ABSTRACT

A main weakness of the Massive Open Online Learning movement is retention: a small minority of learners (on average 5-10%, in extreme cases <1%) that start a MOOC complete it successfully. There are many reasons why learners are unsuccessful, among the most important ones is the lack of self-regulation: learners are often not able to self-regulate their learning behavior. Designing tools that provide learners with a greater awareness of their learning is vital to the future success of MOOC environments. Detecting learners' loss of focus during learning is particularly important, as this can allow us to intervene and return the learners' attention to the learning materials. One technological affordance to detect such loss of focus are webcams—ubiquitous pieces of hardware available in almost all laptops today. Recently, researchers have begun to make use of webcams as part of complex machine learning-based solutions to detect inattention or loss of focus based on eye tracking and eye gaze data. However, those approaches tend to have a high detection lag, are inaccurate, and are complex to design and maintain. In contrast, in this paper, we explore the possibility to make use of simple metrics such as gaze presence or face presence to detect a loss of focus in the online learning setting. To this end, we evaluate the performance of three consumer and professional eye-tracking frameworks using a benchmark suite we designed specifically for this purpose: it contains a set of common xMOOC user activities and behaviours. The results of our study show that already those simple metrics pose a significant challenge to current hard- and software-based eye-tracking solutions.

IUI'18, March 07–11, 2018, Tokyo, Japan

ISBN 123-4567-24-567/08/06…$15.00

DOI: http://dx.doi.org/10.475/123_4

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous; See http://acm.org/about/class/1998/ for the full list of ACM classifiers. This section is required.

Author Keywords

Eye-tracking; Online learning; MOOCs.

INTRODUCTION

Massive Open Online Courses (MOOCs) have gained a lot of popularity over the past years and are now being offered to millions of learners on various platforms such as Coursera, Udacity and edX, among others. One major motivation behind MOOCs is the provision of ubiquitous learning to learners across the world and thus making knowledge available for a large and diverse population, increasing their levels of expertise in a wide variety of subjects. Yet, despite their popularity, MOOCs suffer from low levels of learner engagement and learner retention, as only a very small percentage of learners who start a course actually complete it successfully (on average 5-10%, in extreme cases <1%) [8].

One reason why learners fail to complete MOOCs can be found in the design of the platforms. They tend to be rather basic (as a large amount of effort goes towards maintenance) which makes the delivery of the courses not always overly engaging. This contributes to the lack of self-regulation (in planning, motivation, goal setting) learners tend to exhibit, especially those without a higher education background [4]. Here, loss of focus (during video watching, quiz submissions, etc.) is a core challenge, as it can have disastrous effects on learning efficiency [21]. Therefore, interventions which detect this and can guide the learner's focus back to the course content could be of great value. However, detecting the loss of a learner's attention *in real-time* is difficult. Especially in the MOOC environment, there is little measurable feedback from the learner which could be leveraged. As a potential solution, the most promising recent approaches employ ubiquitous consumer-grade webcams to "observe" a learner during their activities in the MOOC platform. In a brief prelimary study in

one of our MOOCs, we found that around 1/3 of the learners would indeed be willing to use such camera-based technology. As approaches that continuously record a learner's face and environment in their entirety are too complex, invasive, and questionable from a privacy-perspective, client-side gaze-tracking (i.e. only the gaze—instead of the face—is recorded and all data is directly processed on the learner's machine) seems to be a more most suitable technique. Here, the eye-mind-link [17] is exploited as the eye gaze usually correlates well with a person's focus. However, previous approaches [1, 2, 14, 27] are hampered by two problems: the extremely high delay between a loss-of-focus event and its detection (usually 30-60 seconds), and low levels of detection accuracy [27].

In this paper, we explore a significantly simpler alternative approach towards detecting a loss of focus whilst learning in a MOOC environment: detecting the departure of a user's face from the webcam's viewport as a proxy for her stopping to pay attention to MOOC (video) content—a user whose face is not in front of the screen is unlikely to pay attention to a video playing on it. However, even this deceptively simple detection task is challenging in a MOOC environment using only consumer-grade hard- and software. Therefore, in this paper we conduct an extensive study involving two popular browser-based software frameworks for gaze and eye detection, `tracking.js` and `WebGazer.js`. Those frameworks can both be potentially integrated into current MOOC environments, and perform all their processing on the user's computer without the need for backend server infrastructure or additional software installations. We benchmark the ability of those frameworks to reliably detect a user's face (in the following, called *face-hit* and *face-miss* for detecting or not detecting a face) in a variety of common MOOC user activities (e.g., watching a MOOC video while leaning on one hand, checking something on a smartphone, drinking coffee, etc.), and under different environmental conditions (e.g., wearing glasses or not, different lighting conditions, different backgrounds).

The following research questions are in the focus of our study:

RQ1 Which activities are relevant to a MOOC learner's behavior which might affect their face positioning in front of the screen? To this end, we compile a list of typical activities, with their expected duration's and expected influence on face detection.

RQ2 How reliable can current software frameworks detect face-hit and face-miss events under typical MOOC conditions? Here, we conduct an extensive lab study involving two open-source consumer webcam frameworks and a professional eye tracker as a baseline, with 20 study participants performing the aforementioned tasks in a controlled environment. Unfortunately, we will show that current software and hardware technology still struggles to provide consistent high detection quality for these tasks.

RELATED WORK EYE TRACKING FRAMEWORKS

Attention Loss in Learning

Different data collection methods have been used to study the loss of attention or focus of students in traditional classrooms since the 1960s, such as the observation of inattention behaviors [7], the retention of course content [13], using direct probes in class [22, 10] and relying on self-reports from students [3]. A common belief was that learners' attention may decrease considerably after 10-15 minutes of the lecture, which was supported by [22]. However, Wilson and Korn [26] later challenged this claim and argued that more research is needed. In a recent study, Bunce et al. [3] asked learners to report their attention loss voluntarily during 9-12 minute course segments. Three buttons were placed in front of each learner, representing attention lapses of 1 minute or less, of 2-3 minutes and of 5 minutes or more. During the lectures, the learners were asked to report their loss of attention by pressing one of three buttons once they *noticed* their attention loss. This setup led Bunce et al. [3] to conclude that learners start losing their attention early on in the lecture and may cycle through several attention states within the 9-12 minute course segments.

In online learning environments, losing attention may be even more frequent. Risko et al. [18] used three one hour video-recorded lectures with different topics (psychology, economics, and classics) in their experiments. While watching the videos, participants were probed four times throughout each video. The attention-loss frequency among the participants was found to be 43%. Additionally, Risko et al. [18] found a significant negative correlation between test performance and loss of attention. Szpunar et al. [23] investigated the impact of interpolated tests on learners' loss of attention within online lectures. The study participants were asked to watch a 21-minute video lecture (4 segments with 5.5 minutes per segment) and report their loss of attention in response to random probes (one probe per segment). In their experiments, the loss of attention frequency was about 40%. Loh et al. [11] also employed probes to measure learners' loss of attention and found a positive correlation between media multitasking activity and learners' loss of attention (average frequency of 32%) whilst watching video lectures. Based on these considerably high loss of attention frequencies we conclude that reducing loss of attention in online learning is an important approach to improve learning outcomes.

Inspired by the eye-mind link effect [17], a number of previous studies [1, 2, 14] focused on the automatic detection of learners' loss of attention by means of gaze data. In [1, 2], Bixler and D'Mello investigated the detection of learners' loss of attention during computerized reading. To generate the ground truth, the study participants were asked to manually report their loss of attention when an auditory probe (i.e. a beep) was triggered. Based on those reports, the loss of attention frequency ranged from 24.3% to 30.1%. During the experiment, gaze data was collected using a dedicated eye tracker. In [14], Mills et al. asked the study participants to watch a 32 minute, non-educational movie and self-report their loss of attention throughout. In order to detect loss of attention automatically, statistical features and the relationship between gaze and video content were considered. In contrast to [1, 2], the authors mainly focused on the relationship between a participant's gaze and areas of interest (AOIs), specific areas in the video a participant should be interested (like the speaker

or slides). In [27], Zhao et al. present a method for detecting inattention similar to the studies in [14], but adapted and optimized it for a MOOC setting.

All mentioned approaches relying on the eye-mind link share two common flaws: they are usually unable to provide real-time feedback as they are trained on eye-gaze recordings with sparse manually provided labels (e.g., most approaches have a label frequency of 30-60 seconds, which directly translates into a detection delay of similar length), and the reported accuracy is too low for practical application (e.g., [27] reports detection accuracy of 14%-35% depending on training and video). As a result, we choose a different approach as discussed in the following sections.

Eye Tracker Frameworks

In this paper, we rely on eye tracking systems to detect if a MOOC learner is indeed looking at the screen or not. We include two types of systems: a hardware-based eye tracker and software-based eye trackers. As a hardware-based system, we use the professional high-end eye tracker Tobii X2-30 Compact[1], which has for example been used in academic works to evaluate marketing stimuli [9], but also for understanding learning processes [20]. This eye tracker costs around 6000 Euro and is thus unsuitable for scalable MOOC deployment. However, we use it as a high-quality baseline. `Tobii` uses its own proprietary analytic software Tobii Studio.

As hardware for the software-based solutions, we use the built-in camera of our experimentation laptop, a Dell Inspiron 5759 with a 17-inch screen and a 1920×1080 resolution. To estimate the gaze points based on a live webcam feed, we relied on `WebGazer.js` [16][2], an open source eye tracking library written in JavaScript. webgazer can be configured with different components for tracking gaze, pupils, or faces. We used two such components: the first is *clmtrackr*[3], a face fitting library (referred to as `CLM` in the following), which has been used in academic works for selfie analysis [25], camera-based emotion detection [19], or intelligent public displays in city environments [15]. `CLM` tracks a face the coordinate positions of a face model, as for example shown Figure 1. Using this face model, Webgazer can extrapolate the user's gaze (i.e., the point of the screen on which a user's gaze focuses) by estimating the face's distance and orientation from the screen. Unfortunately, the `CLM` API of Webgazer only allows access to the extrapolated gaza data, and not the face model itself, which restricts our experimental design (see below). An additional, much more severe weaknesses of `CLM` is that the face-fitting algorithm can be very aggressive, even when no face is present. This leads to many potential problems where random background elements (like posters, plants, furniture) are mistaken for faces, and sometimes even preferred over a real user's face clearly visible in the camera's viewport. This problem is also reflected in the low performance in our studies in the result section.

[1] `https://www.tobiipro.com/product-listing/tobii-pro-x2-30/`

[2] `https://webgazer.cs.brown.edu`

[3] `https://github.com/auduno/clmtrackr`

Figure 1. Face fitting model used by Clmtrackr; also shows a common fitting error due to hand positioning

As a simpler alternative, we also use the *tracking.js* [12][4] face recognition library (`TJS` in the following), which has been employed for example in security systems [6] or for general object recognition tasks [24]. With respect to eye and face tracking, this library offers a significantly less powerful feature set than both `Tobii` and `CLM`, as it can only detect the presence and location of the boundary box of faces in a video stream (see Figure 2). While it can also be used to track the location of eyes (but not the gaze), we did not use that feature in this study. We hypothesize that the simplicity of `TJS` leads to more reliable and stable face-miss event detection.

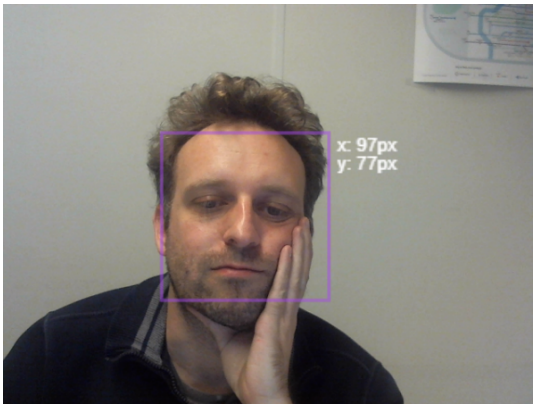

Figure 2. Face Boundary Boxes in tracking.js

The differences in nature of the three used frameworks leads to different heuristics for detecting a `face-miss` event, i.e., detecting when a user's face turns or moves away from the computer screen.

- `Tobii`: A `face-miss` event is detected if the proprietary Tobii Studio software cannot determine gaze point coordinates. This usually represents a problem with detecting the users' eyes by the tracker hardware (e.g., they are not within the camera viewport, they are closed, or obstructed by an object). Sometimes, while the eyes can be found by the Tobii eye tracker, still no gaze coordinates can be determined as the gaze direction is unclear. We cannot distinguish this

[4] `https://trackingjs.com`

case from a case where there is no face at all. In our experience, the presence of gaze coordinates is a very reliable proxy for the presence of a face (low false positive rate), while the lack of coordinates does not necessarily imply the absence of a face.

- `CLM`: Similar to `Tobii`, we define a `face-miss` event as the software's inability to fix exact gaze coordinates. In contrast to `Tobii`, due to the aggressiveness of the face fitting algorithm, `CLM` is is quite prone to detect faces where in reality, there are none (high false positive rate).

- `TJS`: We define a `face-miss` event as the library's inability to fix a face boundary box in the webcam's video stream. Here, we do not try to track eyes or gaze.

The video or eye tracker stream is continuously processed while it is recorded. The `Tobii` system relies on dedicated hardware support for this task (which partially contributes to its high retail price), and is thus able to guarantee a sampling rate of 30 samples per second mostly independent of the user computer hardware. For the webcam-based solutions, image processing of the video stream needs to be handled by the system's CPU in the browser's environment. As a result, only low sampling rates are possible without overwhelming low-end computer systems, and we decided on a fixed sampling rate of 4 samples per second. However, due to the unreliability of the JavaScript timer events under high system loads, the standard deviation of the targeted sampling time of 250ms is 48ms in our experiments (described further in Section User Study). Furthermore, we have extreme cases where the sampling times increased up to 1157ms, i.e., less than one sample per second. Therefore, `Tobii` should be able to react with significantly lower delays than the webcam-based frameworks.

USER STUDY

In order to evaluate the suitability of the chosen webcam toolkits for face and gaze tracking, we developed a **benchmark set of tasks**, which we argue represent common behaviours of online learners in front of their laptops. For each of the tasks we define the desired behaviour: the eyetracking devices should either report the loss of the face/gaze (in the case of `face-miss` tasks) or keep detecting the face/gaze (in the case of `face-hit` tasks). We exclude mobile learners from these tasks as desktop learners are still the vast majority of learners in today's MOOC environment. More concretely, among the more than twenty MOOCs our institution offers on the edX platform less than 20% of learners access the course content via mobile devices.

We designed a total of fifty tasks together with a small sample of regular MOOC learners (graduate students in our research lab) that are—to some extent—abstract versions of the behaviour MOOC learners exhibit when watching lecture videos, one of the most common activity in so-called xMOOCs (i.e. MOOCs that are heavily relying on video lectures to convey knowledge, in contrast to cMOOCs which rely on learners' self-formed communities and peer teaching). The task descriptoins are shown in Table 3.2 in the appendix. They fall under three broad categories:

- `face-miss` tasks contain those user behaviours that should result in the eyetracker's loss of face/gaze detection. Twenty-one tasks belong to this category; examples include *Take a sip from the cup [next to you] while turning away from the camera* or *Look straight up to the ceiling for 8 seconds*.

- `likely-face-miss` tasks should result in eyetrackers reporting a mix of face hit and face miss samples. Two examples among the fourteen tasks in this category are *Lean back and put your hands behind your neck for 5 seconds* and *Draw a square on the paper*.

- Lastly, `face-hit` tasks describe user behaviours that should not influence the eyetrackers' ability to detect the face, but may influence gaze detection. Fourteen tasks belong to this category, for example *Reposition yourself in the chair* and *Stare at the camera for 3 seconds*

We developed a Web application that included both webcam-based eyetrackers and presented the twenty tasks as cue cards to the study participants in the browser[5]. We conducted the user study on a Intel i7, Windows 10 laptop which has a builtin webcam situated in the center of the top screen bezel. Next to the two software eyetrackers we also equipped the laptop with the Tobii X2-30 eyetracker, a professional hardware eyetracker which was placed on the lower screen bezel.

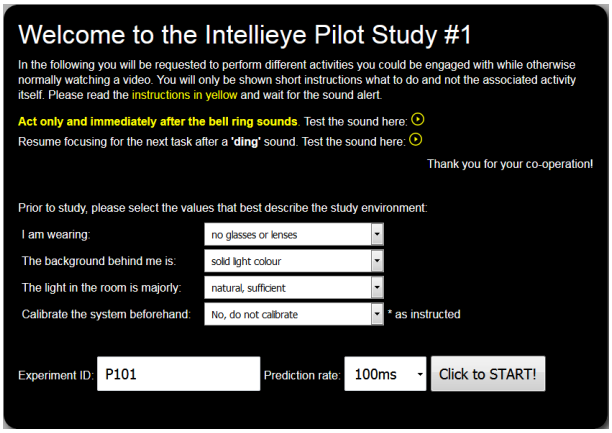

Figure 3. Opening screen of the user study

The opening screen of the application is shown in Figure 3. For each study participants the fifty tasks were shown in a randomized order (an example task cue card is shown in Figure 4). The procedure for each task Q_i is the same: the task description is shown and five seconds later a bell sound indicates the start of the task at time $t_{start}^{Q_i}$: at the sound of the bell the participant is expected to perform the task. Another bell sound (different to the one indicating the start) indicates to the participant when the task has been finished at time $t_{end}^{Q_i}$, and this is followed by the next task description. Task durations differ, depending on the specific task, e.g. Q31 requires a

[5]We designed the application in a modular manner; additional eyetracking frameworks can easily be evaluated as well. We have open-sourced our application at our companion webpage [5].

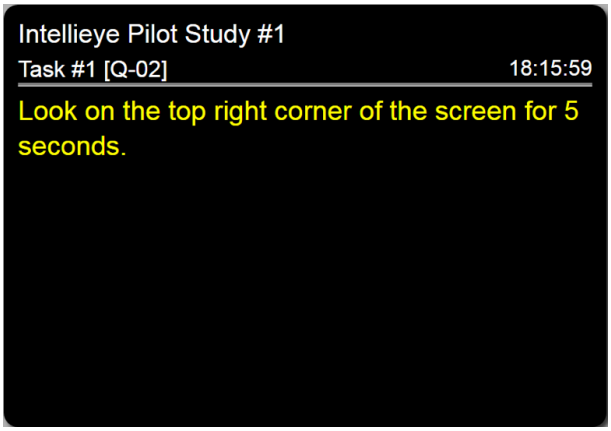

Figure 4. Example task "cue card" of the user study.

participant to look at a certain angle for 5 seconds while Q39 asks a participant to check his or her phone for 10 seconds.

As this is a controlled study, in order to facilitate the proper execution of the tasks, the participants were provided with the necessary tools to perform all tasks, including a sheet of paper and a pen (required for Q22, Q24 & Q25), a cup (Q41 & Q42) and a phone (Q39).

The `Tobii` requires a calibration step which participants concluded at the start of the study. The `CLM` eyetracker can also be calibrated (light-weight: five red dots are shown on the screen that have to be clicked one after the other). To test the effect of the calibration we randomly switched on the calibration step for eight of the twenty learners.

To prepare the participants for the tasks, each participant was trained on two tasks before the start of the actual study. The participants were reminded repeatedly to only start executing a task after the sound of the bell and to keep executing them until the ending sound occurred.

Study participants

The study was conducted across a one week period: twenty participants were recruited among the graduate students and staff members of a large European university via email lists. The participants did not receive any compensation and spent less than an hour on this study. Among the twenty participants, nine wore glasses and two had contact lenses. In ten of the sessions the background behind the test subject had a uniform (light) color, in another 10 cases a poster or photographic background was observed. We recorded these settings in our study as we had conducted preliminary experiments which indicated that eye-trackers (especially the software-based ones) can be mislead by noisy backgrounds.

Detection accuracy

For every task and participant, we determine the eyetrackers' face-hit/face-miss predictions from the collected logs inbetween the $t^{Q_i}_{start}$ and $t^{Q_i}_{end}$ timestamps. As the eyetrackers vary in their sampling rate (cf. Section 2.2) they all produce a differing amount of labels (face-hit,face-miss) for each sample interval. We evaluate the accuracy of the produced labels by computing the percentage of correct predictions (as defined by the type of task) in the task interval. As an example, in a five second task slot the webcam-based sample once every 250m (on averag), and thus we collect approximately 20 predictions. For a `face-miss` task, if 14 out of the 20 predictions predicted a miss, the detection accuracy will be 70%. Lastly, we average the accuracies for each task across all participants.

Table 1. `Tobii`'s delay between the start of a face-miss/likely-face-miss task and the first face-miss event. The data is averaged across all participants of a single task.

Delay	% of tasks
1 sec	53%
2 sec	28%
3 sec	6%
4 sec	3%
5+ sec	9%

Table 2. Overview of the impact of the participants' background on `TJS`'s and `Tobii`'s accuracy.

Background	#	Accuracy in % `TJS`	`Tobii`
Solid light	10	61.5	68.6
Poster/photo	10	55.7	67.8

RESULTS

Performance

The first question we consider is the accuracy of the three eyetrackers under investigation across our fifty tasks. Table 3.2 shows detection accuracy for each task, aggregated across our twenty study participants. As expected, `Tobii` achieves the highest accuracy, with an average of 68.2% across all tasks. Among the two Webcam-based eye-trackers `TJS` clearly outperforms `CLM`, achieving an average accuracy of 58.6% compared to `CLM`'s 35.4%. If we were only to focus on the tasks where face misses and likely face misses form the ground truth, `CLM`'s accuracy would drop to 9.6%. The reason for this poor performance is `CLM`'s approach to face and gaze detection: it will try to match anything in the video frame to a potential face area, a separate face detection phase is not performed. This also explains its high accuracies in the face hits tasks. Note that the calibration step performed by some of our participants for `CLM` also did not result in a different outcome.

The comparison between `Tobii` and `TJS` shows a relatively small performance gap between the Webcam-based eye-tracker and the high-end device. While `Tobii` outperforms `TJS` in 39 of the 50 tasks, in many instances the difference in accuracies is rather small. Using `Tobii` as a reference point, `TJS` is able to conform with 77.8% of `Tobii`'s detected labels.

Due to the clear performance differences between `TJS` and `CLM`, in the further analyses we focus exclusively on `TJS` and its performance compared to `Tobii`.

Reaction Times

As one of the potential reasons for `TJS`'s lag in performance compared to `Tobii` we investigated the reaction times of both

users and frameworks. More specifically, we measured the delay between the *instructed* start time of the task (i.e., the timestamp $t_{start}^{Q_i}$) and the first time a library detects a face-miss. This time delta of course consists of both the user delay (i.e., the time it take for the user to finally start performing the task, which for some tasks—e.g. Q23 & Q46—showed a considerable delay) and the actual detection delay imposed by the framework. We averaged the delays of all participants for a particular task and report the percentage of tasks whose average delay is up to 1 second, up to 2 seconds, etc. in Table 1. For the majority of tasks, the high-end device is able to detect the first face-miss within a second of the start of the task.

The `Tobii` eye tracker runs with a very high fixed sampling rate of 30 samples per second, and is mostly unaffected by the current CPU load of the host machine. It is guaranteed to react without noticeable delays from the manufacturers side. Therefore, we make the assumption that the delays in Table 1 represent the user delay. In contrast, `TJS` and `CLM` can have very low sampling rates depending on the current system load (we aim at 4 samples per second, but we have reports of significantly lower rates). By comparing the times of detecting the first face-miss of both `TJS` and `CLM` with `Tobii`, we can obtain an intuition of the delays imposed by those frameworks. For `TJS`, this resulted in a delay of 0.6 ± 1.1 seconds, and for `CLM` in 1.3 ± 1.0 seconds. While these detection delays are not instantaneous, the delays are short enough for practical applications and far from the delays of 30-60 seconds reported in e.g. [27], where a machine learning pipeline was trained to detect learners' loss of focus during video watching.

Background as an Influencing Factor

As we conducted the user study in different rooms on different times of the day, we also recorded our participants with various backgrounds. In Table 2 we partitioned our participants according to the background they sat in front of during the study. All participants reported their background to be either of a solid light color (as present in many offices) or contain a poster and/or photo. This factor had an impact on the eyetrackers' accuracy: while `Tobii`'s accuracy remained unaffected by the background, the `TJS` eye tracker considerably degraded when a noisy background was introduced.

SUMMARY AND DISCUSSION

In this paper, we have examined the challenge of detecting the presence of a user's face in front of a computer screen while performing typical MOOC-related activities. This can serve as an approximation for a future real-time attention and inattention detection mechanism for MOOC environments, stimulating and supporting self-regulated learning. We compared three popular potential technical solutions for this task: using the high-quality and high-price professional `Tobii` hardware eye tracker, and using two software-based solutions relying on analyzing the video stream of a consumer-grade Webcam. Two open-source libraries where used for this task, the gaze tracking library *Clmtrackr* and the face tracking library *tracking.js*. We conducted an extensive user study with 20 participants, who had to perform a controlled benchmark suite of 50 realistic tasks where their face was either in front of the tracking device or not, introducing several challenging influence factors like body movement, partially covering the face, noisy backgrounds, or crooked body postures. This benchmark suite allows for a standardized and fair comparison of even vastly different approaches for `face-hit` and `face-miss` detection, and we provide it under an open-source license for fostering future research in this area on our companion Web page [5].

Our experiments showed that the professional dedicated hardware solution outperforms the open-source software-based solutions both with respect to detection performance and processing speed, but is of course unsuitable for a large-scale deployment outside of a controlled lab setting. For the software-based solutions which can indeed run on typical hardware used by MOOC learners, the unnecessarily complicated gaze tracking as employed by `CLM` introduces many complications, resulting in poor detection performance both for the presence and absence of a user's face. In contrast, the more simplistic face tracking library `TJS` shows significantly higher performance for nearly all benchmark tasks. Additionally, both software libraries incur an additional time delay of around 1-2 seconds over the nearly instantaneous detection response of the hardware solution. With careful design, this delay should be easily manageable in a future MOOC learner attention detection component.

As future steps, we indeed plan an implementation of such an attention detector suitable for a large-scale MOOC deployment on the basis of the `TJS` framework. Beyond purely technical or methodical challenges, this allows approaching many additional interesting research questions: Would MOOC learners be willing to accept and use such an attention detection tool? What are the reasons why they would like/or refuse to use such technology? And of course finally, if learners accept the use of such tools, does this indeed positively impact their learning performance?

REFERENCES

1. Robert Bixler and Sidney D'Mello. 2014. Toward fully automated person-independent detection of mind wandering. In *UMAP'14*. Springer, 37–48.
2. Robert Bixler and Sidney D'Mello. 2016. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction* 26, 1 (2016), 33–68.
3. Diane M Bunce, Elizabeth A Flens, and Kelly Y Neiles. 2010. How long can students pay attention in class? A study of student attention decline using clickers. *Journal of Chemical Education* 87, 12 (2010), 1438–1443.
4. Dan Davis, Ioana Jivet, René F. Kizilcec, Guanliang Chen, Claudia Hauff, and Geert-Jan Houben. 2017. Follow the Successful Crowd: Raising MOOC Completion Rates Through Social Comparison at Scale. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference (LAK '17)*. 454–463.
5. Removed for Anonymity. 2017. Companion Webpage for this paper with additional information and materials. -. (2017).

6. Barbara Hauer. 2016. Continuous Supervision: A Novel Concept for Enhancing Data Leakage Prevention. In *European Conference on Cyber Warfare and Security*. Academic Conferences International Limited, 342.
7. Alex H Johnstone and Frederick Percival. 1976. Attention breaks in lectures. *Education in chemistry* 13, 2 (1976), 49–50.
8. Katy Jordan. 2014. Initial trends in enrolment and completion of massive open online courses. *The International Review of Research in Open and Distributed Learning* 15, 1 (2014).
9. Rami N Khushaba, Chelsea Wise, Sarath Kodagoda, Jordan Louviere, Barbara E Kahn, and Claudia Townsend. 2013. Consumer neuroscience: Assessing the brain response to marketing stimuli using electroencephalogram (EEG) and eye tracking. *Expert Systems with Applications* 40, 9 (2013), 3803–3812.
10. Sophie I Lindquist and John P McLean. 2011. Daydreaming and its correlates in an educational environment. *Learning and Individual Differences* 21, 2 (2011), 158–167.
11. Kep Kee Loh, Benjamin Zhi Hui Tan, and Stephen Wee Hun Lim. 2016. Media multitasking predicts video-recorded lecture learning performance through mind wandering tendencies. *Computers in Human Behavior* 63 (2016), 943–947.
12. Eduardo Lundgren, Thiago Rocha, Zeno Rocha, Pablo Carvalho, and Maira Bello. 2015. tracking. js: A modern approach for Computer Vision on the web. *Online]. Dosegljivo: https://trackingjs. com/[Dostopano 30. 5. 2016]* (2015).
13. John McLeish. 1968. *The lecture method*. Cambridge Institute of Education.
14. Caitlin Mills, Robert Bixler, Xinyi Wang, and Sidney K D'Mello. 2016. Automatic Gaze-Based Detection of Mind Wandering during Narrative Film Comprehension. In *EDM'16*. 30–37.
15. Masaki Ogawa, Takuro Yonezawa, Jin Nakazawa, and Hideyuki Tokuda. 2015. Exploring user model of the city by using interactive public display application. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*. ACM, 1595–1598.
16. Alexandra Papoutsaki, Nediyana Daskalova, Patsorn Sangkloy, Jeff Huang, James Laskey, and James Hays. 2016. WebGazer: scalable webcam eye tracking using user interactions. In *IJCAI'16*. 3839–3845.
17. Keith Rayner. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin* 124, 3 (1998), 372–422.
18. Evan F Risko, Nicola Anderson, Amara Sarwal, Megan Engelhardt, and Alan Kingstone. 2012. Everyday attention: variation in mind wandering and memory in a lecture. *Applied Cognitive Psychology* 26, 2 (2012), 234–242.
19. Filipe Santos, Ana Almeida, Constantino Martins, Paulo Moura de Oliveira, and Ramiro Gonçalves. 2017. Hybrid Tourism Recommendation System Based on Functionality/Accessibility Levels. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*. Springer, 221–228.
20. Amir Shareghi Najar, Antonija Mitrovic, and Kourosh Neshatian. 2015. Eye tracking and studying examples: how novices and advanced learners study SQL examples. *CIT. Journal of Computing and Information Technology* 23, 2 (2015), 171–190.
21. Jonathan Smallwood, Daniel J Fishman, and Jonathan W Schooler. 2007. Counting the cost of an absent mind: Mind wandering as an underrecognized influence on educational performance. *Psychonomic bulletin & review* 14, 2 (2007), 230–236.
22. John Stuart and RJD Rutherford. 1978. Medical student concentration during lectures. *The lancet* 312, 8088 (1978), 514–516.
23. Karl K Szpunar, Novall Y Khan, and Daniel L Schacter. 2013. Interpolated memory tests reduce mind wandering and improve learning of online lectures. *Proceedings of the National Academy of Sciences* 110, 16 (2013), 6313–6317.
24. Sajjad Taheri, Alexander Veidenbaum, Alexandru Nicolau, and Mohammad R Haghighat. OpenCV. js: Computer Vision Processing for the Web. (????).
25. Sara Tedmori and Rashed Al-Lahaseh. 2016. Towards a selfie social network with automatically generated sentiment-bearing hashtags. In *Computer Science and Information Technology (CSIT), 2016 7th International Conference on*. IEEE, 1–6.
26. Karen Wilson and James H Korn. 2007. Attention during lectures: Beyond ten minutes. *Teaching of Psychology* 34, 2 (2007), 85–89.
27. Yue Zhao, Christoph Lofi, and Claudia Hauff. 2017. Scalable Mind-Wandering Detection for MOOCs: A Webcam-Based Approach. In *European Conference on Technology Enhanced Learning*. Springer, 330–344.

Table 3. Overview of all fifty benchmark tasks, and the accuracy (in %) of CLM, TJS and Tobii averaged across the 20 participants in our user study.

QID	Task	Accuracy in % CLM	TJS	Tobii
	FACE MISS Tasks			
Q1	Cover the camera for 2 seconds	12	**45**	7
Q2	Cover the camera for 5 seconds	28	**73**	17
Q3	Cover your face with both hands for 5 seconds	17	67	**75**
Q4	Look what is under your table (3 sec)	3	64	**81**
Q5	Stand up for 5 seconds	10	68	**71**
Q20	Tilt your head to the right for 3 seconds	15	**59**	38
Q21	Check if there is a HDMI port on the laptop	12	56	**77**
Q26	Look straight up to the ceiling for 8 seconds	12	72	**92**
Q27	Tilt your head back for 5 seconds (face ceiling)	10	68	**84**
Q28	Tilt your head back for 2 seconds (face ceiling)	5	51	**66**
Q29	Look down for 3 seconds	4	35	**78**
Q32	Look left for 2 seconds	7	50	**72**
Q33	Look left for 8 seconds	14	69	**88**
Q35	Look over your right shoulder	13	50	**72**
Q36	Look right for 10 seconds	13	77	**90**
Q37	Look right for 3 seconds	14	64	**79**
Q38	Look right for 5 seconds	7	63	**83**
Q39	Check your phone for 10 seconds	7	42	**89**
Q40	Check your phone, return after the ding	13	37	**87**
Q42	Take a sip from the cup while turning away from the camera, return it after the ding	5	40	**51**
Q47	Look up and return immediately	8	49	**68**
	LIKELY FACE MISS Tasks			
Q6	Lean back and put your hands behind your neck for 5 seconds	2	**67**	63
Q7	Lean closer to the screen and immediately back	3	17	**27**
Q13	Rapidly lean back and forth until the ding sounds	6	37	**57**
Q18	Tilt your body to the left and stay for 3 seconds	13	50	**57**
Q19	Tilt your body to the right and return immediately	6	41	**55**
Q22	Draw a square on the paper	9	45	**67**
Q23	Write down 5 keys left from letter A, focus back to the screen only after the ding	4	19	**61**
Q24	Write down a sentence about weather	15	47	**73**
Q25	Write down *I love Intellieye!*	10	45	**78**
Q30	Look half-left and return	7	36	**64**
Q31	Look half-right and stay for about 5 seconds	7	42	**77**
Q41	Face the camera and take a sip from the cup until you hear the ding	8	30	**35**
Q46	Cover the left side of your face with left hand over cheek and eye	8	38	**43**
Q48	Look around in the room to every direction	10	63	**82**
	FACE HIT Tasks			
Q8	Open browser and navigate to www.weather.com. Return after the ding. (15sec)	94	**97**	80
Q9	Open new browser tab and return to this after the ding	**95**	89	87
Q10	Open some program window (e.g. My computer) on top of study window and return after the ding	**99**	87	94
Q11	Feeling sleepy? Yawn and cover your mouth with a hand. (3 sec)	**94**	66	64
Q12	Grab the tip of your nose for 3 seconds	**100**	64	71
Q14	Reposition yourself in the chair	**98**	77	61
Q15	Scratch the top of your head (or nape) for 3 seconds	**94**	69	85
Q16	Scratch the lower part of your left leg for 2 seconds	**93**	79	64
Q17	Slowly lean back and stay for about 2 seconds	**96**	32	38
Q34	Look on the top right corner of your screen for 5 seconds	95	86	**96**
Q43	Rest your eyes for 5 seconds (close them)	**95**	84	14
Q44	Scratch your left cheek for 3 seconds	**95**	74	89
Q45	Sit still and face the camera for 5 seconds	**94**	87	90
Q49	Grab your ears with both of your hands for 3 seconds	**95**	76	85
Q50	Stare at the camera for 3 seconds	**95**	89	88