A Concept to Analyze User Navigation Behavior Inside a Recorded Lecture
(to Identify Difficult Spots)

Matthias Bauer and Christoph Meinel
Hasso Plattner Institute (HPI)
University of Potsdam
Potsdam, Germany
email: {matthias.bauer,christoph.meinel}@hpi.de

Abstract—This paper provides a brief report of our concept to scan the streaming server’s log files in order to identify specific behavior of the users. A distinct form of behavior is the jump-back. Students do it when they watched a scene of a recorded lecture and then watch it again after a short amount of time. So, it can be assumed that this scene is of higher interest because it is either very interesting or hard to understand for the viewer. The knowledge of these found hotspots could be used in order to improve teaching materials such as slides and teaching style. In this paper, we describe how we plan to gather the data, how to analyze it and how the insights can be utilized. It is not only focused on the technological perspective of video-based e-learning but also on the pedagogical view.

Keywords—E-Learning; Video Lecture; Log Analysis; Learning Analytics

I. INTRODUCTION

Hasso Plattner Institute (HPI) is an institute at the University of Potsdam in Germany. It has around 400 students doing their Bachelor or Master degree in IT Systems Engineering and 10 professors. There are three lecture halls and several seminar rooms where lectures and presentations take place on a daily basis. Around 20 lectures held at HPI are recorded with our tele-TASK recording system every week. That makes around 30 hours of new material every week [1].

In order to let the students consume the recorded lectures, there is a web portal [11] where all the recordings are being published. Students can watch them online or download the smaller podcast episodes to watch them offline on their mobile devices for example when they are on the bus or train. Our web portal offers several functions for the students to participate more actively for example features to rate, annotate, tag and link lectures as well as an innovative search function that - among static metadata - is also based on automatic speech recognition (ASR) and optical character recognition (OCR) analysis results. More general information about the tele-TASK project and research topics can be found at [2] and [3].

The structure of this paper is as follows: after this introduction, we give a short overview of related work and why our approach does have a purpose. In Section 3, we describe our plans of realizing the technical aspect and in Section 4, we discuss the meaning of the insights we can collect and how they could be used in order to improve video-based e-learning.

II. RELATED WORK

We found highly interesting research [4] on the detection of disengagement in e-learning environments. They analyze the log files of the HTML learning system they utilize. Back [5] on the other hand, concentrates on the analysis of Facebook data while they have a look at the learning behavior of students being abroad. Bruckman [6] analyze log file data in order to get an understanding of the behavior of learners in a distinct learning community of their MOOSE crossing virtual learning environment. In their paper [7], other researchers intend to help the teachers with the discovery of patterns in learner’s actions through log file analysis but they do not explain the structure and origin of their log files which seem to be manually generated observations of collaborative learning behavior.

None of the above mentioned sophisticated approaches focuses on video-based e-learning and so they do not intend to analyze the log files of a streaming server delivering the learning material but rather focus on websites’ log files or other retrieved log data. Their methods and results are very interesting and contribute good insights to the field but there is still the need for research in the field of video lectures and the resulting wish for analyzing the learner’s behavior.

There has been earlier research from our group. The study in [8] focuses on changes in the interest of learners while using the a web portal for video-based e-learning. In order to do so, HTTP server logs and Helix streaming server logs are being analyzed. We have developed a new recording system that produces lecture videos in MP4 containers encoded with the video codec H.264/AVC and the audio codec Advanced
Audio Coding (AAC). These videos are being delivered by an Adobe Flash Media Server [10]. A Helix server that has been used for several years, will be shut down soon because no new RealMedia content will be produced in the future in our institute. So, there is a need to develop a new solution focusing on the current infrastructure. Besides, our goal is different from the work in [8] where the focus was on the learner’s interest whereas here our focus is on hotspots - scenes that are watched again within a short amount of time.

Figure 2. Workflow of recording a lecture.

III. MOTIVATION AND PLAN

With this paper, we intend to set a starting point for ongoing research with the goal of improving the quality of video learning material. We do not want to start a survey among the learners and have them fill out a questionnaire concerning the quality and their personal experience with a specific video lecture because on the one hand side this does not seem to be very efficient because there is a lot of manual work to do and on the other hand side we would not like to bother the students but rather have an automatic workflow that is capable of handling mass data. The tele-TASK web portal which is the media library for all the recorded lectures has around 2000 unique visitors per week [9] and we can expect that most of them are visiting to watch lectures online. So, there is an adequate number of users to produce log data with their individual behavior. Learners watch online lectures with the tele-TASK video player, an online player that was built to fit the special needs and requirements. In Figure 3, it is shown that two videos are played back simultaneously and there are additional navigation tools, such as chapter marks and automatically detected slide previews, which also can be used as links to a distinct position inside the video lecture.

Figure 3. Example of the video player showing a lecture.

A. Users Seeking Back Cause Jump-Back Events

What we want to look at first are jump-back events. As a jump-back event we categorize every seeking behavior inside a video lecture within a small time frame. The small amount of time shall sort out every occurrence of watching the video again on another day. Finding the spots where users jump back in a video requires log file analysis. As all the video content is delivered by a streaming server, we have to analyze its log files as opposed to many other solutions analyzing the log files of a web server.

B. Possible Interpretations of Found Jump-Back Events

A user seeking to an earlier position he or she has already watched causing a jump-back event might have done this out of several reasons. One explanation is that this spot in the video was very interesting and the student watches it again not to miss anything. Another reason for watching it again – and unfortunately, this is quite often the case – is that this topic was too hard to understand during the first time watching it. This is not unusual, especially taking into account that we are looking at lectures at a university. So the topics are supposed to be difficult. And yet there should be a way for the lecturers to identify those topics in their lectures that are the most difficult. Also, it is possible that simply the teaching speed is too fast for the learners. Let us assume that the students theoretically could understand the taught topic. But when the teaching speed is much faster than the learning speed of the students they cannot memorize everything by watching the lecture and every contained scene only once.

Sometimes, it is necessary that teachers rephrase what they explain or give an example to illustrate the topic or at least find a way of making the topic a little bit easier to understand for their students. We think that the identification of clusters of jump-back events can be a tool to do so.

C. False-Positive Jump-back Events

We still have to consider that some of the jump-back events are caused by other reasons. Beside those mentioned ones like watching a difficult topic, bad explanation or too fast teaching speed, we have to take a look at the setting the learner finds himself or herself in: in most cases our learners watch the lectures at home. So, there is a variety of disturbing factors such as family members or roommates, ringing telephones, emails or chat messages popping up, etc. Or the student simply takes a break. After any of these kinds of disruption, it is very likely that the learner seeks back in the lecture in order to watch the last scene again.

There are two reasons why these false-positive events are not that fatal: on the one hand side, there will not be an accumulation of false-positive events at a distinct point of a lecture because the reasons of the disruptions are individual and non-systematic. So with an increasing number of views of an online lecture, the impact of false-positive events will decrease and the number and validity of jump-back event accumulations will increase. On the other hand side, we can sort out some of the false-positive events by defining a maximum time between the client-pause and seek-back operations. Because the longer the elapsed time the more likely a disruption was the reason for the jump-back event.
D. The Ideal Jump-Back Window Size

In order to get results as accurate as possible, we have to think about the ideal window size of the observed elapsed time (server clock). If the window is too small, we might accidentally detect the seeking operations of a user searching for something inside the video. So the elapsed time should be at least 10 seconds (not to forget, we are looking at jump-back events following play events, only)

One the other hand side, if the window is too big, we might detect non-content related disruptions of the learning process as mentioned earlier.

Actually, there is a second window that has to be optimized: the time of the jump-back length. For example, if a user jumps from 15:20min to 14:40min, the jump-back length is 40 seconds. Here, we can optimize almost in the identical way with the first one: If the chosen window is too small, we will detect normal seeking behavior while searching for a distinct spot. If the window is too big, we might either catch disruptions where the learner did not click on the pause button or we might catch repetitions in the learning process, for example if a learner watches the lecture, or a bigger part of it, again. We intend to find difficult scenes, so our focus is on smaller units rather than whole lectures.

IV. INTENDED TECHNICAL REALIZATION

Before we are going to outline our idea, we would like to give a brief overview of the server infrastructure.

A. Technical Overview

First of all we should mention that the tele-TASK project is based on several servers. There is an Apache webserver with a MySQL database to host the website which is the lecture video library. In addition, there are a transcoding server and an analysis server that are used to provide automatic conversion into other video formats and generation of podcasts and analysis tasks such as slide detection, OCR and ASR, respectively. The distribution of the lecture videos is done by an Adobe Flash Media Server (FMS), currently in version 4.5.6. The FMS has a logging function that is able to create log entries for nearly all kinds of detectable atomic events. These events are for example server-start, connect, publish, play, pause, seek and many more. For our purpose seek events are the most interesting ones among the events caused by user interaction with the video (player). In the log files, they are combined with lots of other columns.

The most important ones for our purpose are: c-client-id, x-event, x-suri or x-file-name, x-spos or c-spos and x-page-url. The c-client-id is an identifier that helps us distinguish different users. Privacy concerns do not have to be taken into account because Adobe FMS creates unique but non traceable identifier strings. IP addresses, client software or other user-related data are not logged. x-event is the action that was logged. Only events of type seek are important for us here. We can distinguish different lecture videos with the columns x-suri and x-file-name. For finding a jump-back event we have to find two log entries where the user and video are the same but there is a seek event seeking to an earlier time within the same video. This should be limited to a small time window to exclude non-expedient occurrences where the jump-back does not happen right after watching the distinct scene. In order to identify the time position we need to have a look at x-spos which is the streaming position in milliseconds but also at c-spos which contains the client stream position when a “client-pause” or “client-seek” event is logged. Also, we have to exclude all the entries with an x-page-url containing the URL of the development/test server which can be identified easily by the finding the string “www5-dev” there.

B. Workflow

The analysis of the retrieved log data should work with a minimal amount of human intervention. The main steps are:
(which symbolizes the elapsed time in the video) at an earlier time of day from the same client watching the same video.

The legitimate question why we do not use an existing analysis framework such as Sawmill [13], can be answered as follows. Even though there are quite a few professional products with manifold functions, we could not find a solution capable of analyzing Adobe FMS logs according to our special requirements, for example compare the log file entries considering the position fields x-spos and c-spos.

D. Visualization

All of the found hotspots inside the lectures should be announced to the respective lecturer only. So he or she will not feel exposed to the public or offended by a public display of some kind of “automatic quality measurement” of their respective work. This should be done with a close connection to the video player so that it is easier and faster to find the intended scene. We already have a display of automatically detected slides as a kind of preview within the video player as shown in Figure 3. These preview slides are shown in a kind of clustered timeline. The shown clusters symbolize the display time and duration of a distinct slide. This previously done work can be adapted to show the identified hotspots to the lecturer of the presentation. The lecturer will have to be logged in to the web portal so that according to the lecturer’s account with unique login data the video player can be enabled to show the found results.

The visible feedback must be self-explanatory and realizable with standard web programming components or frameworks. In a first iteration for early demonstration purposes, we are going to mark the extracted key frames of the slides video with a surrounding red box. In this way we would define our hotspot resolution to be the display duration of a distinct slide (a “scene”). So the lecturer can see which scenes in their presentation are hotspots in terms of clusters of jump-back events. Depending on the duration a distinct slide is shown, this could be very inaccurate (in case a lecturer shows a single slide for lots of minutes or he or she does not use the slide as a holder of significant information). So the first iteration of visualizing the hotspots should only be temporary and should be replaced by a more accurate and informative way where the lecturer can see the number of jump-back events and their exact positions (not depending on the slide but on the actual position in the lecture).

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed an idea of how to analyze the log files of a streaming server that delivers recorded lectures. We focus on jump-back events because we think that an accumulation of these can serve as an indicator for either a very interesting spot or a topic or explanation that is too difficult to understand at the first time. Further, we propose how these insights can be utilized and made available to the lecturer or author of this distinct lecture or learning video. So it might help the lecturers to receive anonymous feedback which can be used to improve their teaching.

In future work, we have to finish implementing the analysis engine and realize the feedback component for the lecturer so that they can see the hotspots easily and draw their own conclusions. The feedback should be visible to the lecturer and make it very easy to find the specific spot within the lecture. Here we can orient ourselves towards the annotation and keyword functions in the web portal. They already provide a way to create clickable links that can open a lecture in the video player and jump automatically to the desired position in both videos (the one showing the lecturer and the other one the slides), simultaneously.

In a further step, we would like to use our implementation and experience for HPI’s massive open online courses (MOOC) at OpenHPI [12]. They have a different learning online platform especially customized for their necessities but they also use the tele-TASK video player. Here, we expect even more and more significant results because there we have less and shorter videos and more users.

VI. REFERENCES