Smart Jump: Automated Navigation Suggestion for Videos in MOOCs

Han Zhang
Department of Computer Science and Technology
Tsinghua University
hanzhan15@mails.tsinghua.edu.cn

Maosong Sun
Department of Computer Science and Technology
Tsinghua University
sms@tsinghua.edu.cn

Xiaochen Wang
Department of Computer Science and Technology
Tsinghua University
xiaochen15@mails.tsinghua.edu.cn

Zhengyang Song
Institute of Interdisciplinary Information Sciences
Tsinghua University
songzy16@mails.tsinghua.edu.cn

Jie Tang*
Department of Computer Science and Technology
Tsinghua University
jietang@tsinghua.edu.cn

Jimeng Sun
Computational Science and Engineering at College of Computing
Georgia Institute of Technology
jsun@cc.gatech.edu

ABSTRACT

Statistics show that, on average, each user of Massive Open Online Courses (MOOCs) uses "jump-back" to navigate a course video for 2.6 times. By taking a closer look at the navigation data, we found that more than half of the jump-backs are due to the "bad" positions of the previous jump-backs. In this work, employing one of the largest Chinese MOOCs, XuetangX.com, as the source for our research, we study the extent to which we can develop a methodology to understand the user intention and help the user alleviate this problem by suggesting the best position for a jump-back. We demonstrate that it is possible to accurately predict 90% of users' jump-back intentions in the real online system. Moreover, our study reveals several interesting patterns, e.g., students in non-science courses tend to jump back from the first half of the course video, and students in science courses tend to replay for longer time.

Keywords

MOOCs; user intention; video navigation

1. INTRODUCTION

MOOCs boom swiftly in recent years and have attracted millions of users worldwide. This is not only transforming higher education, but also provides fodder for scientific research [20, 10]. Thanks to MOOC platforms, we are able to track all students' behaviors and outcomes in such a fine-granularity that were never available before. For example, we know when they watched a lecture and how they watched a lecture (e.g., in normal speed or faster pace, any pause or rewind), their patterns in answering the quiz questions (e.g., how long it took them, what their answers were), and the ultimate outcomes (e.g., certificate completion and final course score). Such activity log data became the collective memory of the education experiences from millions of students and teachers. Analyzing such data from MOOCs provides novel and great potential for understanding students' behaviors and enhancing education delivery.

Meanwhile, despite the vast number of students being reached by MOOCs, the overall success of MOOCs as the delivery vehicle for future education is still in debate. One major challenge of MOOCs is how to design "smart" interactions to improve student engagement. For example, the overall course completion rate is still very low—i.e., 5% or lower [15, 18]; and only a small portion of students complete the course video watching. One reason is due to the lack of deeper interactions between the system and the users (students). The system does not offer a mechanism to understand user intentions so as to provide necessary help in users' studying procedures.

Watching course videos probably is the most important activity in MOOCs. Actually, the majority of time that students spend on MOOCs is watching videos [2]. Recently, a few researches have been conducted on the click-level interactions between users and the MOOC systems in order to better understand how users learn and what they need when watching video [7, 8, 11, 14, 23]. The studied interactions include playing video, pause, jump-back (rewind), or jump-forward, etc. However, we soon found that the jump-back is a frequent behavior with strong user intention. Our preliminary study shows that, on average, each MOOC user clicks "jump-back" 2.6 times to navigate a course video. The reasons may vary a lot including there is some difficult part that the user cannot understand, and the user simply missed some part for other reasons. One interesting question arises: can we leverage the emerging AI techniques to help alleviate this problem?

Employing one of the largest Chinese MOOCs, XuetangX.com, as the source for our research, we try to conduct a systematic study on this problem. We found that more than half of the jump-backs are due to the "bad" position of the previous jump-backs. Because they usually try several times to find the correct position. Our goal then is to study the extent to which we can develop a mechanism

*corresponding author

WWW'17 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054166

to understand user intentions and help user locate the best position for a jump-back. To precisely illustrate the problem we are going to deal with, we give an example scenario in Figure 1.[1] The user is watching a lecture of the most popular course on XuetangX.com, i.e., *Financial Analysis and Decision Making*. She is trying to jump back and the system automatically detects her intention and suggests four positions to go. For example, the place with the highest probability (0.35) is to describe an example in the lecture, and the second position is the concept of "capital assets". The dark green curve shows the jump-back navigation distribution of this lecture according to the historical data. The problem is referred to as *automated navigation suggestion*. We see that the challenge here is how to design a principled suggestion methodology by considering the lecture content, personal preferences of the current user, and historical navigation behavior of all users who watched this lecture.

We develop an algorithm based on deterministic finite automaton to reconstruct users' jump-back behaviors. To capture the content information, users' preferences, and historical navigation behaviors, we propose a data-driven method which combines general behaviors and personal preferences for jump-back prediction. To summarize, the main contributions of this work include:

- We formally define an interesting problem of automated navigation suggestion in MOOCs, and systematically study the problem on a real large MOOC dataset.
- Our study on the dataset reveals several interesting phenomena. For example, students in non-science courses tend to jump back from the first half of the course video, and students in science courses tend to replay for longer distance.
- We propose a predictive method to predict users' jump-back behaviors, and demonstrate that it is possible to accurately predict 90% of users' jump-back intentions in the real online system.

Organization. Section 2 reviews the related work. Section 3 introduces the dataset and Section 4 presents our analyses on the dataset. Section 5 formulates the problem and presents the proposed methodology. Finally, Section 6 presents experimental results and Section 7 concludes this work.

2. RELATED WORK

We review related literature in three aspects: understanding user behaviors, enhancing user engagements and facilitating video navigation in MOOCs.

2.1 Understanding User Behaviors

This line of research mainly focuses on studying user behavior patterns and their implications. For example, Kim et al. [14] identify five student behavior patterns that can explain interaction peaks in video: starting from the beginning of a new material, returning to missed content, following a tutorial step, replaying a brief segment, and repeating a non-visual explanation. Li et al. [16] cluster user behaviors into patterns (the cluster centers) based on numerical features of available interaction types, i.e., pausing, forward and backward seeking and speed changing. Furthermore, to facilitate the analysis of user behaviors, Chorianopoulos et al. [9] present a system that builds user activity graphs based on user video browsing actions to help understand the video content. Based on user behavior analysis, some researchers also find strong correlations between user behaviors and video content. Chorianopoulos et al. [7] find that interesting video segments can be detected through users' collective interactions (e.g., seek/scrub, play, pause) with the video. Following this work, Avlonitis et al. [1] find that users' replaying activity significantly matches the important segments in information-rich and visually complex videos. Related studies can also be found in [24, 22]. These works inspire us on selecting useful video interaction activities from the chaotic log data. However, most of these works aim at analyzing or visualizing data, rather than studying interaction data in depth to facilitate video navigation.

[1]This function will be deployed online soon.

Figure 1: Personalized jump-back suggestion for a specific user. The distribution of possible end positions for a jump-back is shown above the navigation bar. The red circles with different size represents possible end positions, where a larger circle means a larger probability. We also show the context transcripts around each red circle to illustrate the content.

2.2 Enhancing User Engagements

Researchers have found that video production affects user engagement to some extent [12], since user interacting experience with video depends on the style of video production. As a result, some user-friendly video productions have been designed to enhance user engagement. Pongnumkul et al. [17] propose a content-aware dynamic timeline control which decouples video speed and playback speed to deal with the problem that video frames flash too fast when sudden jumps occur. Uchihara et al. [19] present a hierarchical prefetching asynchronous streaming technology that considers a video-scene structure to improve user interacting experience when the network speed is slow. To deal with the same problem, Carlier et al. [4] propose a different approach whereby the prefetch hit rate is improved by biasing the users' seeking to prefetched content, through changing the video player user interface. A new video interaction model presented in [6] helps people quickly browse videos with predefined semantic rules from their preliminary result of user study. Our work contributes the current research about enhancing user engagement by providing automated navigation suggestions when users jump back on video progress bar.

2.3 Facilitating Video Navigation

Several previous works focus on the specific research of video navigation in MOOCs. For example, Yadav et al. [21] design a system that provides nonlinear navigation in educational videos, which utilizes features derived from a combination of audio and vi-

Table 1: the Description of the Dataset

Course	Category	Type	Number
Science	Video	Total #	791
		Avg. length	303.71
	User	Total #	26,487
		Max #users/course	12,989
		Min #users/course	7,590
	Complete-jump	Total #	112,854
		Max #Cjs/course	52,939
		Min #Cjs/course	27,316
Non-science	Video	Total #	438
		Avg. length	635.28
	User	Total #	8,598
		Max #users/course	5,126
		Min #users/course	1,540
	Complete-jump	Total #	7,569
		Max #Cjs/course	2,802
		Min #Cjs/course	2,012

sual content of a video. Carlier et al. [5] collect viewing statistics as users view a video, and use these data to reinforce the recommendation of viewports for users. Kim et al. [13] present a 2D video timeline with an embedded visualization of collective navigation traces and a visual summary representing points with frequent learner activity. To the best of our knowledge, there was little work combining video content and user preferences to predict and suggest positions in timeline for video navigation.

3. DATASET

3.1 Data Description

Our study is based on the data from XuetangX, one of the largest MOOC platform in China. XuetangX, being in partnership with edX, was launched in October 2013 and up to now, it has offered 808 courses and attracted more than 5,900,000 registered users. These courses cover various fields like computer science, economics, business, electronics, art, history, etc. The dataset used in this study includes six courses from XuetangX which are opened during Sep. 2015 to Dec. 2016—i.e., *Financial Analysis and Decision Making* (FAD), *Data Structure* (DS), *Principle of Circuits* (PC), *Japanese Language and Culture* (JLC), *the Aesthetics of Modern Life* (AML), *Chinese Ancient Civilization Etiquette* (CACE). These courses can be categorized into two types: science (Computer Science, Electronic Engineering and Economics) vs. non-science (Language, Art and Culture) courses. The number of lecture videos of each course varies from 49 to 489. The length of each video is ranging from 1 minute to 30 minutes. Overall, the number of registered users for the science courses is larger than that of the non-science courses. Table 1 lists statistics of the dataset.

3.2 Data Preprocessing

Now we introduce how we preprocess the dataset. To begin with, we first give the definition of the concept **Complete-jump** (Cj).

Definition 1. ***Complete-jump.*** *A complete-jump consists of one (or multiple) jump-back actions by the same user on the same lecture video, aiming to find the right position to rewind. We use the tuple (u, v, p_s, p_e) to denote a complete-jump that user u jumps back from start position p_s to end position p_e in video v.*

Please note that in the definition, the complete-jump may consist of multiple jump-back actions, which means that the user may jump back to a position of no interest and continue to seek for the position that she wants to replay. This also implies that in a complete-jump behavior, there might be a jump-forward action. For example, the user jumps back far away and then wants to jump forward a bit to correct it. We keep all the video interaction related actions, including playing video, pausing, jumping and stopping video. The complete-jump behavior actually cannot be obtained straightforwardly. We need to figure out which complete-jump behavior a specific jump-back action belongs to, or similarly how many jump-back actions construct a complete-jump behavior. We design an algorithm based on deterministic finite automaton to reconstruct them. First, we introduce three atomic events of which a complete-jump consists:

Jumping back (Jb): When a user drag the current cursor (p_s) or click on the progress bar of the video to go to a previous position (p_e) (p_e<p_s), then we say there is a jumping back event.

Jumping forward (Jf): When a user drag the current cursor (p_s) or click on the progress bar of the video to go to a position afterwards (p_e), i.e., (p_e>p_s), then we say there is a jumping forward event.

Short watching (Sw): After jumped to a position in the video, the user usually would take a look (for seconds) to see whether this position is the desired one. We call this event as a short watching event. Each Sw has a duration period between two jumping events, i.e., from t_1 to t_2, where t_1 is the time at which the first jumping event occurs and t_2 the next jumping. The duration should be no longer than t_{max} (t_2-$t_1 \leq t_{max}$). In our experiments, we tentatively set t_{max}=10 seconds.[2]

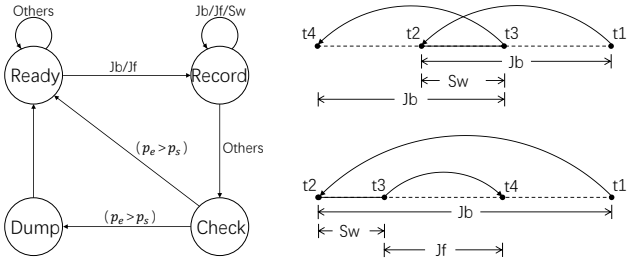
Based on the above defined events, we use a deterministic finite automaton (DFA) to construct the complete-jump behaviors. Figure 2a illustrates the state transition in the DFA. In total, there are four states: Ready, Record, Check, Dump. When the state is Ready, it stays until receives a Jb or Jf event, then the state transforms to Record. At the Record state, it maintains a stack. When getting the Jb, Jf and Sw events, it pushes all the events into the stack. If there comes some other events, it goes to Check state. At the Check state, we check p_s of the start position of the event in the bottom of the stack, and p_e of the end position of the event at the top of the stack. If p_e>p_s, we say that the sequence of events in the stack constitute a jump-forward behavior, which is not the focus of this work. Then the state goes back to Ready. Otherwise, the state goes to Dump, where we aggregate the sequence of events in the stack to construct a complete-jump behavior.

Figure 2b shows two basic complete-jump patterns. The first pattern illustrates a kind of complete-jumps that consist of the event sequence [Jb, Sw, Jb], which means that the user jumps back to a position and watches from this position for a few seconds, and then jumps to an earlier position. The second pattern shows a complete-jump that consists of the event sequence [Jb, Sw, Sf]. In this kind of scenario, the user jumps to an early position firstly and then jumps forward to a later position. The two patterns are basic patterns and can constitute many other patterns by combining together. For example, the complete-jump consists of the event sequence [Jb, Sw, Jb, Sw, Jb] is the combination of two pattern 1s, and the complete-jump consists of the event sequence [Jb, Sw, Jb, Sw, Jf] is the combination of a pattern 1 and a pattern 2.

4. OBSERVATIONS

In this section, we engage in some high-level investigation of the factors that cause users to jump to different positions in a video.

[2]We tried different setting for $t_{\max}$ and empirically selected 10 as an optimal setting.



(a) complete-jump construction (b) Two basic complete-jump patterns

Figure 2: The left figure illustrates the construction of complete-jump behavior based on DFA and the right figure shows two basic complete-jump patterns. In the first pattern (top), the user jumps back, watches the video for some time, then jumps back again, and in the second one (bottom), the user jumps back, watches the video and jumps forward.

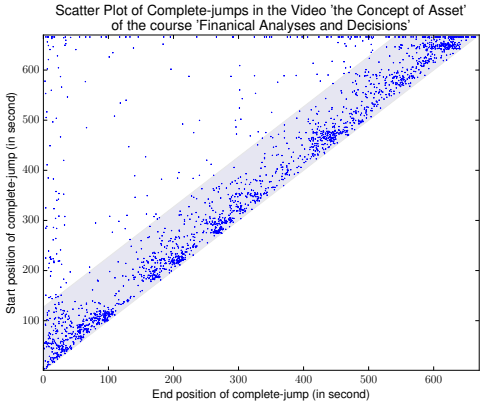


Figure 3: **All complete-jumps in a specific video. Horizontal and vertical axes represent the timeline of the video. A spot at (x, y) represents a complete-jump from position y to position x in video timeline. The blue area contains 90% spots in the figure.**

The investigation is conducted from two perspectives: **(1) General performances:** what is the general performance when users jump back? How does the general performance vary in different courses and videos? **(2) User preferences:** Does users have personal preferences when they jump back? Strong or weak? When or how they show their preferences?

4.1 General performances

To provide a clear view of users' general performance in a video, we plot all the complete-jumps of a selected video in Figure 3. A spot (x, y) in Figure 3 represents a complete-jump from position y to position x in the timeline of the video. We see that most spots are close to the diagonal. It means that users usually do not rewind far away from the current position. We call the time duration between the start position and end position as **jump span**. In this example, the jump span of 90% complete-jumps are smaller than 230 seconds, shown as the blue area. This phenomenon also exists in other videos.

Now we describe the investigation results about the differences of general performances in two levels of granularity:

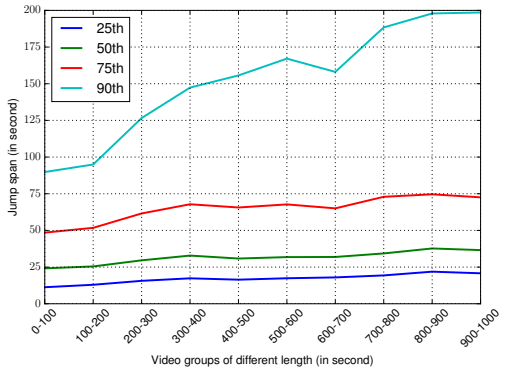


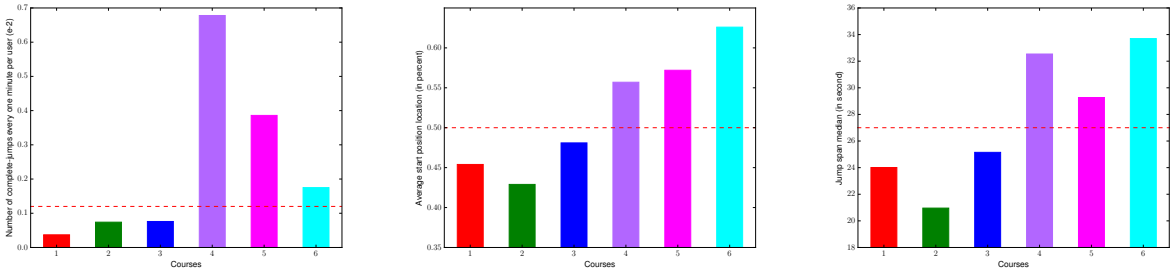
Figure 5: **The distribution of different percentiles for jump span in different video length groups. Y-axis: position of jump span percentiles. X-axis: video groups of different length.**

By course. What are the differences of the general jump-back patterns between different courses? Figure 4 shows the results from three aspects: degree of complete-jump activity, average start position location and jump span median. Of the six courses in Figure 4, the first three are non-science courses (JLC, AML, ACE), and the other three are science courses (FAD, PC, DS). Figure 4a shows the degree of complete-jump activity, which is counted as the average number of complete-jumps per minute per user of all the videos in one course. We see that the complete-jump activity of science courses is much more active than that of non-science courses. This probably is because science courses are relative more difficult than non-science courses. Users need to study some videos multiple times in order to understand the content. Figure 4b is the average start positions in percentage of video length for each course. It shows that users in non-science courses tend to rewind from the first half of a video, while users in science courses tend to minimize the rewind and only jump back to previous part of the video. Figure 4c shows the jump span medians for each course. The jump span medians of non-science courses are less than 26 seconds while the jump span medians of science courses are more than 28 seconds. It indicates that users of science courses are likely to rewind farther than those of non-science courses.

By video. The lengths of different videos vary a lot. We are interested in knowing whether the length of a video has effect on the complete-jump behavior. Figure 5 shows the correlation between video length and jump span. We group the videos by length in every 100 seconds, and examine the distribution of jump span in different video groups. The distribution is represented by its 25th, 50th, 75th and 90th percentile of jump span. The results in figure 5 can be summarized as follows: (1) jump span is positively correlated with the length of videos. (2) the effects of video length are different on complete-jumps with short and long jump span. Complete-jumps with longer jump span are more easily to be affected by video length.

4.2 User preferences

Different individual users would have different jump-back patterns. For example, patient users are likely to reply from a farther away position than impatient users. In order to catch users' preferences, we categorize users into different types based on their jump span records leveraging k-means clustering.



(a) Degree of complete-jump activity (b) Average start position location (c) Jump span median

Figure 4: General complete-jump performance comparison in different courses. Y-axis: (a) the number of complete-jumps per minute per user of all videos in one course, (b) average start position location in percent of each course, (c) jump span median of each course. X-axis: the label of six courses.

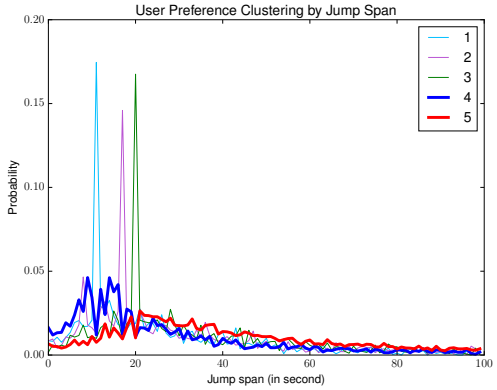


Figure 6: **Cluster users into five categories by their complete-jump records. Y-axis: the probability of different jump spans. X-axis: jump span (in second). Category 1, 2, 3 represent users that have obvious preference, while category 4 and 5 represent users that almost have no preference.**

Figure 6 plots the distributions of the five categories of users based on their jump span records. Category 1, 2 and 3 have sharp peaks in their distributions. It means that users of these three categories have clear preference when they jump back— users of category 1 prefer to jumping back to 10 seconds ago, while users of category 2 and 3 have a preference to jump back farther away (17 seconds and 20 seconds, respectively). Users of the rest two categories (4 and 5) exhibit a relatively flat distribution. Their jump back behavior is more or less unpredictable.

5. AUTOMATED NAVIGATION SUGGESTION

In this section, we formally formulate the automated navigation suggestion problem. We first introduce an approach to segment videos, which is the basis for the following navigation suggestion task. Then, we give the problem definition.

5.1 Problem Formulation

It is not desirable to suggest a position in the middle of two consecutive words, for example a position between two words of a phrase. Before making the suggestion, we first segment each video into snippets using the transcript of each video. Specifically, each

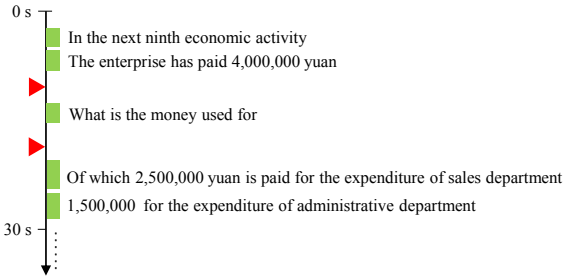


Figure 7: Distribution of transcript sentences (green squares) in the timeline (the vertical line). If the interval time between two consecutive sentences exceed a predefined threshold, a breakpoint (red triangles) will be set there.

video has a corresponding transcript. Each sentence in the transcript is time-stamped with start time and end time. There is also duration time between two sentences, which varies in length among sentences. Figure 7 plots the time distribution of sentences in an example transcript. We set a breakpoint in the middle of two sentences when the duration time between them is longer than a threshold Δt. The threshold is selected according to the following principles: (1) to result in more effective complete-jumps. A complete-jump is called effective when its start position and end position are located in different segments. (2) to result in more non-empty segments. A segment is called non-empty when it contains at least one start position or end position of some complete-jumps. To balance these two principles, we set harmonic mean as the objective function to select Δt:

$$\underset{\Delta t}{\operatorname{argmax}}\ 2\frac{R_{e\_cj}\ \cdot\ R_{n\_s}}{R_{e\_cj}\ +\ R_{n\_s}} \tag{1}$$

where R_{e_cj} is the ratio of effective complete-jumps and R_{n_s} is the ratio of non-empty segments.

After segmentation, a video is partitioned into a sequence of segments $[s_0, s_1, s_2, ..., s_n]$. Now, our problem is to rank those segments and make suggestions to a user when she is planning to jump back. More specifically, when a user click the cursor on the progress bar of video, it triggers a navigation suggestion. Formally, given a video v, a user u, the start position in segment s_i, the objective is to train a model to maximize the probability that user u would jump back to segment s_j of video v, i.e.,

$$\underset{\Theta}{\operatorname{argmax}}\ P(s_j|u, v, s_i; \Theta) \tag{2}$$

where Θ is a set of parameters of the model.

5.2 Feature Representation

Based on the observations in Section 4, we extract features from both user activity logs and video attributes. To summarize, the features used in our approach can be grouped into the following four categories:

Basic features:

- **User id:** identify a specific user in the dataset, each user has a unique id.
- **Start position id:** identify the segment of a video that the start position locates in, each segment has an unique id.
- **End position id:** identify the segment of a video that the end position locates in.
- **Start time:** the corresponding in-video time of the start position.
- **End time:** the corresponding in-video time of the end position.

Video: these features describe the videos' attributes and users' general performances in videos.

- **Video length:** length of the video.
- **Count:** number of complete-jumpsin the video
- **Active degree:** active degree of complete-jumpsof the video, counted as $\frac{n_{cj,v}}{n_{u,v} \cdot t_v}$, where $n_{cj,v}$ is the number of complete-jumps, $n_{u,v}$ is the number of users and t_v is the length of the video.
- **Jump span distribution:** 25th, 50th, 75th, 90th percentile of jump spans in the video

Start position: these features describe users' general performances at the specific start position.

- **Count:** number of complete-jumpsthat start from the position.
- **Max span:** maximum jump span of all complete-jumpsfrom the start position.
- **Min span:** minimum jump span of all complete-jumpsfrom the start position.
- **Median:** median of jump span of all complete-jumpsfrom the start position.
- **Standard deviation:** standard deviation of jump span of all complete-jumpsfrom the start position.
- **Frequent span:** top 3 jump span of all complete-jumps.
- **Entropy:** entropy of jump span: $-\sum_{x \epsilon l} P(x) log_2 P(x)$. We divide the jump span in equal-sized areas. l is the list of time areas, x is one of the time areas and $P(x)$ is the proportion of complete-jumpswhose jump span locate in time area x.

User preferences: these features describe the preferences of users.

- **Count:** number of complete-jumpsthat the user produces.
- **Max span:** maximum jump span of all complete-jumpsthat the user produces.
- **Min span:** minimum jump span of all complete-jumpsthat the user produces.
- **Median:** median of jump span of all complete-jumpsthat the user produces.
- **Standard deviation:** standard deviation of jump span of all complete-jumpsthat the user produces.
- **Category:** user category generated by k-means clustering.

5.3 Suggestion Model

Eq. 2 only gives a general form of the suggestion and it can be instantiated in different ways. In this paper, we tried logistic regression (LR), Support Vector Machines (SVM), and Factorization Machines (FM). It seems that FM works the best on our problem. We now depict how we use FM to make the navigation suggestions. Factorization Machines can leverage the interactions between variables using factorized parameters. This allows us to learn more complex interactions between variables.

For each tuple (u, v, p_s, p_e), we define a set of features and construct a data instance $\mathbf{x}_i$, and compute the suggestion score by:

$$\hat{y}(\mathbf{x}_i) = w_0 + \sum_{j=1}^{d} w_j x_{i,j} + \sum_{j=1}^{d-1} \sum_{j'=j+1}^{d} x_{i,j} x_{i,j'} \langle \mathbf{p}_j, \mathbf{p}_{j'} \rangle \quad (3)$$

where $y(\mathbf{x}_i) \in [0, 1]$ indicates the likelihood of user u jumps to the corresponding position of $\mathbf{x}_i$; $\mathbf{p}_j, \mathbf{p}_{j'}$ are two k-dimensional latent vectors and $\langle \mathbf{p}_j, \mathbf{p}_{j'} \rangle$ models the interactions between variables $x_{i,j}, x_{i,j'}$ with the dot product of the two latent vectors:

$$\langle \mathbf{p}_j, \mathbf{p}_{j'} \rangle = \sum_{l=1}^{k} p_{j,l} p_{j',l} \quad (4)$$

Given this, Eq.(3) can be also rewritten as:

$$\hat{y}(\mathbf{x}_i) = w_0 + \sum_{j=1}^{d} w_j x_{i,j} + \frac{1}{2} \sum_{l=1}^{k} \left[\left(\sum_{j=1}^{d} p_{j,l} x_{i,j} \right)^2 - \sum_{j=1}^{d} p_{j,l}^2 x_{i,j}^2 \right] \quad (5)$$

where $\Theta = \{w_0, w_1, ..., w_d, p_{1,1}, ..., p_{d,k}\}$ are the model parameters. The model has a closed model equation, and can be learned efficiently by the gradient descent method, e.g. stochastic gradient descent (SGD), based on a variety of loss functions, like square loss, hinge loss, etc. Here, we use square loss as loss function and optimize the model parameters by applying L-2 regularization on latent vector parameters to overcome the overfitting problem. The objective function is defined as follows:

$$\mathcal{O}(\Theta) = \sum_{v_i \in V} \left(\hat{y}(\mathbf{x}_i) - y_i \right)^2 + \lambda \sum_{i=1}^{d} \|\mathbf{p}_i\|^2 \quad (6)$$

where λ is a parameter that controls the regularization value. We adopt stochastic gradient descent method to solve the objective function. The partial derivative of $\hat{y}(\mathbf{x}_i)$ with respect to the model parameters can be written as:

$$\frac{\partial \hat{y}(\mathbf{x}_i)}{\partial \theta} = \begin{cases} 1, & \text{if } \theta \text{ is } w_0 \\ x_{i,j}, & \text{if } \theta \text{ is } w_j \\ x_{i,j} \sum_{f=1}^{d} p_{f,l} x_{i,f} - p_{j,l} x_{i,j}^2, & \text{if } \theta \text{ is } p_{j,l} \end{cases} \quad (7)$$

Then the parameters are updated by moving in the opposite direction of the gradient, yielding:

$$
\begin{aligned}
w_0 &\leftarrow w_0 - \eta \cdot 2(\hat{y}(\mathbf{x_i}) - y_i)\frac{\partial \hat{y}(\mathbf{x_i})}{\partial w_0} \\
w_j &\leftarrow w_j - \eta \cdot 2(\hat{y}(\mathbf{x_i}) - y_i)\frac{\partial \hat{y}(\mathbf{x_i})}{\partial w_j} \\
p_{j,l} &\leftarrow p_{j,l} - \eta \cdot \big(2(\hat{y}(\mathbf{x_i}) - y_i)\frac{\partial \hat{y}(\mathbf{x_i})}{\partial p_{j,l}} + 2\lambda p_{j,l}\big)
\end{aligned} \tag{8}
$$

where $\eta \in R^+$ is the learning rate for gradient descent. Given a training dataset, we iteratively update each parameter according to the gradient until convergence or the maximum number of iterations is reached. Then we can obtain the training model parameters $\Theta = \{w_0, w_1, ..., w_d, p_{1,1}, ..., p_{d,k}\}$.

Navigation Suggestion. Based on users' historical data, we use stochastic gradient descent (SGD) to train a factorization machine model and then employ the trained model to make suggestions when a user trigger a potential suggestion (clicking the cursor on the progress bar of a video). We use Eq. 5 to calculate the suggestion score for each segment based on the trained model and then rank all segments according to the scores. Finally, the top ranked segments are suggested to the user.

6. EXPERIMENTS

In this section, we present various experiments to evaluate the effectiveness of the method based on the observations in Section 4. All of the dataset and codes will be publicly available.

6.1 Experimental Setup

Setting. We split the dataset into training and test sets by time. The training set is comprised of data collected from the first 80% period of time of each course, and the test set is comprised of the data collected from the last 20% period of time of each course. For example, the course FAD opens on 2015.10.9 and ends on 2016.2.1. Then we obtain training data ranged from 2015.10.9 to 2016.1.8, and obtain test data ranged from 2016.1.8 to 2016.2.1. So do the other courses.

Generating Negative Samples. In our experiment, a positive sample corresponds to a complete-jump that truly occurs, denoted as (u, v, p_s, p_e). For each positive sample, we generate several negative samples by choosing different p_es. Given a p_s, there is a list of p_es that users have jumped back to, say, $[p_{e,1}, p_{e,2}, \ldots, p_{e,n}]$. While for a specific complete-jump, there is only one target $p_{e,t}$ ($1 \leq t \leq n$) that the user truly jumps back to. In the remaining list of p_es, we randomly select m p_es to generate negative samples and m is a tunable parameter in the experiment.

6.2 Performance Evaluation

The experiment is conducted in two stages: predicting stage and ranking stage. In predicting stage, we predict the probability of each end position from the same start position. In ranking stage, we rank the end positions by their probabilities predicted in the first stage. The result of ranking stage has two usages: (1) evaluate our method by the measurement hits@n, i.e., the proportion of true end position ranked in the top n. (2) suggest users the top 3 end positions for navigation. First, we conduct the predicting experiments with the following models:

Logistic Regression Classifier (LRC): a logistic regression model is trained and used to predict the probability of each end position.

Table 2: Prediction performance of different method on the dataset

Course	Model	AUC	Recall	Precision	F1-score
Science	LRC	72.46	64.28	25.95	37.37
	SVM	71.92	64.06	25.45	36.42
	FM	**74.02**	**68.36**	**27.61**	**39.28**
Non-science	LRC	72.59	72.96	**69.23**	70.69
	SVM	73.52	79.03	68.39	**73.28**
	FM	**73.57**	**79.82**	67.56	72.88

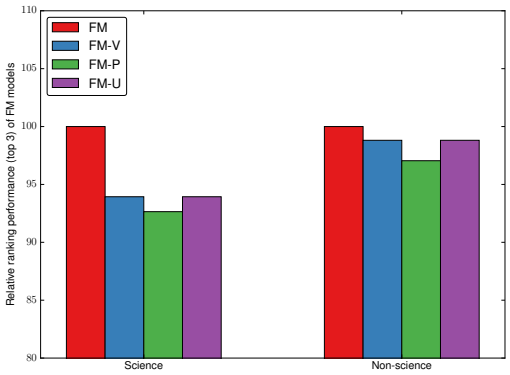


Figure 8: Relative Feature contribution analysis.

SVM: it trains a SVM model to predict predict the probability of each end position.

Factorization Machines (FM): the same features are used to train a FM model and we use it to predict the truly end position of a complete-jump.

We evaluate the performance of different approaches in terms of Area Under Curve(AUC), Recall(Rec.), Precision(Prec.), and F1-Measure(F1) [3]. The result is shown in Table 2.

We can see that the FM model has better performance than other methods. This may be because that there exists correlations among features from videos, start positions and users. The strength of FM just lies in capturing this kind of correlations.

Second, we use the predicting result of FM to compare our automatic suggestion method based on machine learning model with the baseline method based on frequency through a ranking experiment. In predicting experiment, we get the probability of each end position for a given complete-jump. Ranking these end positions by their probabilities produced in the prediction stage, we get an ordered list of them. The true end position of this complete-jump is ranked by a certain order in the list. We use hits@n to measure the suggestion of the true end positions of all complete-jumps. In the method based on frequency, we rank end positions by their frequency for a given complete-jump, and also use hits@n to measure the suggestion of the true end positions of all complete-jumps. Table 3 shows the result of the ranking experiment. We can see that our method based on machine learning model clearly outperforms the method based on frequency both in science courses and non-science courses.

Feature Contribution Analysis. Here we examine the contribution of different categories of features: users' general performances on videos (V), users' general performances at a specific position (P) and the preferences of users (U). First, we use all three categories of features to train a FM model. Then we respectively remove one of the three categories of features, denoted as FM-V, FM-P and FM-

Table 3: Ranking performance of our method based on FM model and baseline method based on frequency with the measurement of hits@n

Course	Method	n = 1	n = 2	n = 3	n = 5
Science	Baseline	33.21	53.21	66.15	81.99
	FM	**37.05**	**60.40**	**76.04**	**89.59**
Non-science	Baseline	39.26	62.61	76.64	91.30
	FM	**42.25**	**72.42**	**88.43**	**96.05**

U. In Figure 8, we regard the performance of all features as 100, and calculate the relative performance of each category of features. We can observe clear drop on the performance when ignoring each category of features. This indicates that our method works well by combining different features and each category of features in our method contributes improvement in the performance.

7. CONCLUSION

Summary. In this paper, we studied an interesting problem of automated navigation suggestions in MOOCs, which aims at deeper interactions between the MOOC systems and users. We use a large collection of data from the courses of Xuetangx.com, providing investigating on jump-back behaviors from different perspectives. We found several interesting patterns and revealed the main factors that influence users' navigation behavior. Based on the discoveries, we developed a methodology aiming to understand the user intention and to suggest the best positions for a jump-back. Our experiments validate the effectiveness of the proposed method. We are also applying the method to a real online system and expect to have the function online very soon.

Future Research. As for the future work, it is interesting to take account of dynamic information, for example, the dynamic behaviors of the user before a jump-back. It is also interesting to explore the influence of visual information of the videos. For the prediction model, it is also helpful to design a better predictive model with higher accuracy.

Acknowledgements. The work is supported by the National Natural Science Foundation of China (61561130160) and a research fund supported by XuetangX.com.

8. REFERENCES

[1] M. Avlonitis and K. Chorianopoulos. Video pulses: user-based modeling of interesting video segments. *Advances in Multimedia*, 2014:2, 2014.

[2] L. Breslow, D. E. Pritchard, J. DeBoer, G. S. Stump, A. D. Ho, and D. T. Seaton. Studying learning in the worldwide classroom: Research into edx's first mooc. volume 8. Research & Practice in Assessment, 2013.

[3] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32. ACM, 2004.

[4] A. Carlier, V. Charvillat, and W. T. Ooi. A video timeline with bookmarks and prefetch state for faster video browsing. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 967–970, 2015.

[5] A. Carlier, G. Ravindra, V. Charvillat, and W. T. Ooi. Combining content-based analysis and crowdsourcing to improve user interaction with zoomable video. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 43–52. ACM, 2011.

[6] K.-Y. Cheng, S.-J. Luo, B.-Y. Chen, and H.-H. Chu. Smartplayer: user-centric video fast-forwarding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 789–798. ACM, 2009.

[7] K. Chorianopoulos. Collective intelligence within web video. *Human-centric Computing and Information Sciences*, 3(1):1, 2013.

[8] K. Chorianopoulos, M. N. Giannakos, and N. Chrisochoides. Open system for video learning analytics. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 153–154. ACM, 2014.

[9] K. Chorianopoulos, I. Leftheriotis, and C. Gkonela. Socialskip: Pragmatic understanding within web video. EuroITV '11, pages 25–28, 2011.

[10] M. M. Crow. Digital learning: Look, then leap. *Nature*, 499(7458):275–277, 2013.

[11] C. Gkonela and K. Chorianopoulos. Videoskip: event detection in social web videos with an implicit user heuristic. *Multimedia Tools and Applications*, 69(2):383–396, 2014.

[12] P. J. Guo, J. Kim, and R. Rubin. How video production affects student engagement: An empirical study of mooc videos. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 41–50. ACM, 2014.

[13] J. Kim, P. J. Guo, C. J. Cai, S.-W. D. Li, K. Z. Gajos, and R. C. Miller. Data-driven interaction techniques for improving navigation of educational videos. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 563–572, 2014.

[14] J. Kim, P. J. Guo, D. T. Seaton, P. Mitros, K. Z. Gajos, and R. C. Miller. Understanding in-video dropouts and interaction peaks inonline lecture videos. In *Proceedings of the First ACM Conference on Learning @ Scale Conference*, L@S '14, pages 31–40, 2014.

[15] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *LAK'13*, pages 170–179, 2013.

[16] N. Li, Ł. Kidziński, P. Jermann, and P. Dillenbourg. Mooc video interaction patterns: What do they tell us? In *Design for Teaching and Learning in a Networked World*, pages 197–210. Springer, 2015.

[17] S. Pongnumkul, J. Wang, G. Ramos, and M. Cohen. Content-aware dynamic timeline for video browsing. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 139–142, 2010.

[18] D. T. Seaton, Y. Bergner, I. Chuang, P. Mitros, and D. E. Pritchard. Who does what in a massive open online course? *Communications of the ACM*, 57(4):58–65, 2014.

[19] N. Uchihara, H. Kasai, Y. Suzuki, and Y. Nishigori. Asynchronous prefetching streaming for quick-scene access in mobile video delivery. *IEEE Transactions on Consumer Electronics*, 56(2):633–641, 2010.

[20] M. Waldrop. Campus 2.0: Moocs are transforming higher education-and providing fodder for scientific research. *Nature*, 14, 2013.

[21] K. Yadav, K. Shrivastava, S. Mohana Prasad, H. Arsikere, S. Patil, R. Kumar, and O. Deshmukh. Content-driven

multi-modal techniques for non-linear video navigation. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, IUI '15, pages 333–344, 2015.

[22] J. Yew and D. A. Shamma. Know your data: Understanding implicit usage versus explicit action in video content classification. In *IS&T/SPIE Electronic Imaging*, pages 78811A–78811A. International Society for Optics and Photonics, 2011.

[23] J. Yew, D. A. Shamma, and E. F. Churchill. Knowing funny: genre perception and categorization in social video sharing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 297–306. ACM, 2011.

[24] B. Yu, W.-Y. Ma, K. Nahrstedt, and H.-J. Zhang. Video summarization based on user log enhanced link analysis. In *ACM MM*, pages 382–391, 2003.