Video Preview Generation Based on Playback Records

Xuewen Shen†, Songlin He*, Dingguo Yu†, and Zhiyan Tang‡
†*Communication University of Zhejiang, Hangzhou, Zhejiang, 310018, China*
{shenxuewen, yudg}@cuz.edu.cn
**New Jersey Institute of Technology, Newark, NJ, 07102, USA*
sh553@njit.edu
‡*Wasu Media & Network Co., Ltd, Hangzhou, Zhejiang, 310018, China*
tangzy8@wasu.com

2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA) | 978-1-7281-8577-4/20/$31.00 ©2020 IEEE | DOI: 10.1109/AICCSA50499.2020.9316530

***Abstract*—A video preview is formed by joining a subset of video snippets from a source video. It is expected to be expressive to help enrich users' experience and hence increase the views of the full video. However, it is challenging to identify suitable video snippets to compose such video previews in support of user recommendation of full-length videos. In this paper, we formulate this problem as an optimization problem to maximize the total playback popularity of video segments based on the analysis of a large amount of users' playback records. We design an algorithm for this problem and provide proof of optimality. Furthermore, we conduct an experiment using real industrial data and recruit volunteers to assess the quality of generated video previews. The experimental results show the feasibility and applicability of our methods.**

***Index Terms*—Video preview, Playback records, Recommender system**

I. Introduction

Video provides a versatile and engaging format of contents for Internet users, and it is predicted that over 82% of the Internet traffic would consist of video (movies, gaming, virtual reality, etc.) by 2022 [1]. Among different video types, short video emerges as a new trend and is experiencing an explosive growth. Examples include TikTok [2], Youtube Shorts [3], Facebook [4], Kuaishou [5]. In fact, the flourishing of the short video market may be a natural phenomenon as people nowadays are embracing a fast-paced lifestyle and would prefer products that could efficiently and precisely provide useful information and therefore save precious time.

In this paper, we focus on *video preview*, which is also a "short-form" video but differs from the regular short video (e.g., videos on Tiktok). A regular short video such as a micro-movie still provide a holistic story though the length is much shorter compared with long videos such as regular movies. On the other hand, a video preview is a subset of video snippets from a source video [6], which reveals only partial but key components of a long video. Generally, media platforms would associate each video with a title, a static thumbnail image, and a textual description to make it attractive to customers. The generation of title and textual description is relatively straightforward, while for the thumbnail, it can be created in multiple ways such as extracting key frames [7] and using deep neural networks [8]. However, these approaches to video display on a website may fall short in precisely conveying the essence of the video content [9]. On the contrary, a (short) video preview, which extracts multiple key segments from the source video, could be potentially more expressive and helpful to enrich users' experience and increase their engagement, thus leading to more views of the full video. A problem, therefore, has naturally arisen: how to automatically generate useful (short) previews for source videos?

There exists a bevy of literature [6], [9], [10] working on different ways to produce video previews. These methods, however, either require intensive computation and therefore put a considerable computational burden on servers for pre-generating and displaying or merely consider the environmental constraints such as devices, networks of producing previews.

In this work, we propose a novel approach to extracting short video previews based on users' playback records. The rationale behind our proposed method is to utilize the playback records left by a set of users for viewed video to produce the most popular segments as video preview, which can later be recommended to newcomers in the video platforms. The more the playback records can be collected from users, the more useful previews can be generated. Remark that the recommender systems (e.g., [11], [12], [13]) typically function on recommending users with several (e.g. top-K) items according to their personal preference. From the perspective of recommendation, our work plays a complementary role to the recommender systems, and to some degree makes the recommendation process more holistic. Overall, our contributions are summarized as follows:

1) We propose a novel approach utilizing users' playback records to generate video preview. Our method is applicable to generate previews for various video types such as movies, micro-videos, user-customized videos, etc.
2) We design the algorithms to produce video previews with the maximum playback popularity (as defined in Section IV), and prove that our solution is optimal.
3) We generate video previews employing real datasets of playback records and conduct experiments to show that the generation is feasible and the video preview is more preferable to users.

The remainder of this paper is organized as follows: we introduce the related work in Section II. In section III, we present the system model and formulate the video preview

978-1-7281-8577-4/20/$31.00 ©2020 IEEE

Authorized licensed use limited to: Miami University Libraries. Downloaded on June 14,2021 at 18:47:35 UTC from IEEE Xplore. Restrictions apply.

generation problem. Section IV provides the detailing algorithms to generate video preview. In section V, we conduct the experiment to show that video preview is more appealing to users compared with static thumbnail images, and Section VI concludes our work and discusses future extension.

II. Related Work

Video preview generation. Some literature [6], [9], [10], [14], [15] have explored the topic of generating video preview. For example, Singh *et al.* [6] proposed to extract the video snippets by extracting and ranking the closed captioning data, events, and associated timestamps. Chen *et al.* [14] investigated the problem of recommending users with key frames of videos by making use of video images and user time-synchronized comments. Our work, from a different perspective, proposes a novel approach of leveraging users' playback records to generate video previews. Meanwhile, we consider the model that the generated video preview can be applicable to other users who do not leave the playback records, while the study in [14] considers the model which recommends to the same individuals in a personalized manner.

Video recommender systems. As stated before, our work can be reckoned as complementary to the video recommender systems, which are typically implemented via collaborative filtering [11], [16], neural networks [12], [17], or reinforcement learning [13]. Our algorithms can take as input the output video items from the aforementioned recommender systems and transfer to the corresponding video previews. Thus forming a more convenient as well as time-saving viewing result for users.

Micro-videos. A similar concept to the video preview is micro-video, which is a new form of user generated contents (UGCs). Many researchers have paid attention to this topic [18], [19], [20]. The micro-video and video preview are both in the form of short video (i.e., around several or tens of seconds) whereas they differentiate with each other as: the former typically provides a holistic content while the latter typically uncovers partial information, i.e., segments containing key frames.

III. System Model and Problem Formulation

In this section, we provide the system model for the short video preview generation and present the problem formulation. We denote with $[n]$ the set of natural integers $\{1, \ldots, n\}$. Meanwhile, the capital letters (e.g., U) always refer to a set while the lower-case letters (e.g., u_i) are specific instances. The key notations related to the system model are listed in Table I for convenient references.

A. System Model

As illustrated in Figure 1, we consider a video platform (e.g., Youtube) has a set of registered users. For these users, $\mathsf{U} := \{u_1, u_2, \ldots\}$ represents those provided playback records for their watched videos while $\mathsf{U}' := \{u'_1, u'_2, \ldots\}$ are those who did not. For each user $u_i \in \mathsf{U}$, the playback records is denoted with $\Gamma_i := \langle v_{ID}, v, \langle t_s, t_e, c\rangle\rangle$ where v_{ID} uniquely identifies the video, and v is the full-length video, and $\langle t_s, t_e, c\rangle$ represents a sequence of video segments, each of which starts at timestamp t_s, and ends at timestamp t_e, and has been replayed c times. All the (full-length) videos along with their playback records from different users, therefore, form a set $\mathsf{V} := \langle (u_i, \Gamma_i)\rangle$. The functionality $\mathcal{F}$ is the core part in our system model taking as input the set V and outputs a new set containing video preview for different videos, denoted as $\mathsf{VP} := \{(v_{ID1}, vp_1), (v_{ID2}, vp_2), \ldots\}$. Moreover, by employing the recommender systems in a black-box manner, users $u' \in \mathsf{U}'$ are recommended with several specific videos of interest. The platform can then find the corresponding video preview from VP via v_{ID} and deliver to target users in U'. Note that u'_i may either be a newcomer to all videos or watched the videos only once[1]. In addition, a user cannot belong to U and U' simultaneously. Once a user $u'_i \in \mathsf{U}'$ plays back a certain video, he or she automatically becomes a member in U in our model.

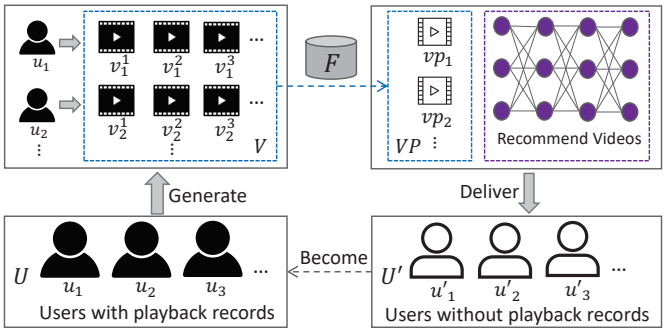

Fig. 1. The system model for video preview generation.

TABLE I
Key notations related to our model.

Notation	Meaning
U	The set of users that provided playback records
U'	The set of users that do not provide playback records
V	The video set containing different users' playback records
VP	The video set for (short) video preview
u_i, u'_i	a specific user in U, U' respectively
n	The number of users who provided playback records for the same specific video
vp	A specific (short) video preview
t_s	The start timestamp of a video segment
t_e	The end timestamp of a video segment
c	The number of times that a video segment replayed
k	The number of video segments in the generated preview vp
Γ_i	The playback records for user i
Δ	The popularity of the video preview vp
T_1	The lower-bound of the length for the generated preview vp
T_2	The upper-bound of the length for the generated preview vp

B. Problem Formulation

Without loss of generality, we consider *one* specific video $v \in \mathsf{V}$ and formulate the problem regarding how to generate

[1]We propose to categorize the users who have watched videos only once into user set U' because i) they do not provide any playback record; and ii) it would be beneficial to both the media platform and users as the former can gain more data of users' preferences and the latter may be interested in watching the videos again.

Authorized licensed use limited to: Miami University Libraries. Downloaded on June 14,2021 at 18:47:35 UTC from IEEE Xplore. Restrictions apply.

(short) video preview vp for such (full-length) v with target function.

Consider that there are $n \in \mathbb{N}$ users $\{u_1, \ldots, u_n\}$ who have generated the playback records for this specific video v. We give the following definition for the playback popularity δ:

Definition 1. Playback Popularity δ**.** *For a certain segment in video v starting from the timestamp t_s and ending at t_e, its playback popularity δ is the total number of replayed times from the n users.*

The δ can essentially quantify the degree of preference to a (potentially short) video segment between t_s and t_e. A segment with a higher δ value is supposed to be accumulated to the final video preview for (other) users. We have the following definition for the target video preview to be generated:

Definition 2. Video Preview vp**.** *For a specific (full-length) video v, its video preview vp is the concatenation of k video segments with the highest playback popularity.*

The parameter $k \in \mathbb{N}$ is the number of video segments in the generated video preview vp, and it decides the final duration of vp, i.e., the preview length $|vp| = \sum_{i=1}^{k}(t_s, t_e)_i$. Note that the length for each segment accumulated in vp may not be equal to each other. We denote with T_1 and T_2 the lower-bound and upper-bound of the length for the generated preview, viz. $T_1 \leq |vp| \leq T_2$. Overall, our problem can be summarized as: for a video $v \in \mathsf{V}$, given all the playback records, i.e., segments with their replayed times, from n users, we need to find k video segments in v, each of which starts from timestamp t_s and ends at t_e, such that:

$$T_1 \leq \sum_{i,i\in[k]} (t_s, t_e)_i \leq T_2 \tag{1}$$

and the objective is to maximize Δ for v, where:

$$\Delta = \sum_{i,i\in[k]} \delta_i \tag{2}$$

Remark that Δ for v is independent with each other in VP, and therefore the maximum of each Δ leads to the maximum of Δ for VP. To simplify the construction of $\mathcal{F}$, we propose to let $\mathcal{F}$ tackles a specific v each time, and repeatedly appends the generated vp to VP. Hence we have the following:

$$\mathsf{VP}_{\max\Delta \ for \ each \ vp\in\mathsf{VP}} \leftarrow \mathcal{F}(\mathsf{V}) \tag{3}$$

Proof of equivalence to continuous knapsack problem. We now briefly discuss how our problem can be mapped, for the specific v, to the *continuous knapsack problem* [21]: First it is obvious that our problem is continuous as each video segment is divisible. Second, the length of each segment $(t_e - t_s)$ maps to the weight of each material to be selected. Third, the playback popularity δ for each segment maps to the value of each material. Last, the constraint of the length of vp maps to the capacity of the knapsack. Additionally, in our problem, the length of vp has both upper-bound and lower-bound while the continuous knapsack problem only requires upper-bound (i.e., the capacity of the knapsack). Our problem needs the additional requirement that the sum of all segments in vp is greater than T_1, thus we can eliminate the constraint of the lower-bound. In this way, our problem to find the vp with a maximum Δ for all the k segments and constrained length can be reduced to the continuous knapsack problem. While from the perspective of *computational complexity*, this problem can be solved in *polynomial time*, we would prove that our solution is optimal in Section IV.

Algorithm 1 $Count()$ - Playback records processing

Input: l, r, Γ
Output: A list Γ' containing a set of segments, each of which is represented by (t_s, t_e, c)

```
Initialize empty lists temp_1, temp_2, Γ'
n = r − l;
Assert n ∈ ℕ;                                  ▷ Ensure n is a positive integer
if n ≡ 1 then
    (v_ID, v, ⟨t_s, t_e, c⟩) ← Γ_0;
    Γ' = (v_ID, ⟨t_s, t_e, c⟩);
else if n ≡ 2 then
    Γ' = (v_ID, Merge(Γ_0, Γ_1));
else if n mod 2 ≡ 0 then
    mid = n/2;                                 ▷ n is an even
    temp_1 = Count(l, mid);
    temp_2 = Count(mid, r);
    Γ' = (v_ID, Merge(temp_1, temp_2));
else
    last = Γ_n;
    mid = ⌊n/2⌋;                               ▷ n is an odd
    temp_1 = Count(l, mid);
    temp_2 = Count(mid, r − 1);
    Γ' = (v_ID, Merge(last, Merge(temp_1, temp_2)));
return Γ'
```

IV. Algorithms Design

In this section, we present the details of generating the video preview for a specific video v based on the playback records from different users.

A. Process Overview

At a high level, the process of recommending users with video previews based on playback records works as follows:

- *Collection*. The users registered in a platform leave playback records for the videos they reviewed. The platform collects these records and stores in a certain data structure (e.g., key-value pairs).
- *Processing*. The processing functionality $\mathcal{F}$ takes as input the (full-length) videos together with playback records and outputs a set containing multiple video previews.
- *Recommendation*. By invoking the functionality of a recommender system in a black-box fashion, the platform can obtain several recommended (full-length) videos for the specific user. Furthermore, the platform can readily query from the video preview set based on the unique video identification, and if available, deliver the corresponding video preview to the users.

B. Design Details

The input for the functionality $\mathcal{F}$ is a set of (full-length) videos V containing playback records from users in U. Again, without loss of generality, we consider generating the video

Authorized licensed use limited to: Miami University Libraries. Downloaded on June 14,2021 at 18:47:35 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2 $Merge()$ - Merging of playback records for two users, and it is a subroutine of Algorithm 1

Input: Γ_1, Γ_2
Output: The merged playback records Γ_3 from two users

Initialize empty lists L_1, L_2, L_3, Γ_3;
$i = j = k = 0$;
$(v_{ID}, v, \langle t_{s1}, t_{e1}, c_1\rangle) \leftarrow \Gamma_1$;
$(v_{ID}, v, \langle t_{s2}, t_{e2}, c_2\rangle) \leftarrow \Gamma_2$;
$l_1 = |\langle t_{s1}, t_{e1}, c_1\rangle|$;
$l_2 = |\langle t_{s2}, t_{e2}, c_2\rangle|$;
for each $item$ in $\langle t_{s1}, t_{e1}, c_1\rangle$ **do**
  $L_1.append((\text{“}s\text{”}, item.t_{s1}))$;
  $L_1.append((\text{“}e\text{”}, item.t_{e1}))$;
for each $item$ in $\langle t_{s2}, t_{e2}, c_2\rangle$ **do**
  $L_2.append((\text{“}s\text{”}, item.t_{s2}))$;
  $L_2.append((\text{“}e\text{”}, item.t_{e2}))$;
while $i < 2 \cdot l_1$ and $j < 2 \cdot l_2$ **do**
  if $L_1[i][1] < L_2[j][1]$ **then**
    if $i \equiv 0$ and $j \equiv 0$ **then**
      $L_3.append((L_1[i][1], 1))$;
    else
      if $L_2[j-1][0] \equiv$ “s” and $L_2[j][0] \equiv$ “e” **then**
        $L_1[i][0] \equiv \text{“}s\text{”} \ ? \ L_3.append(L_1[i][1], 2) : L_3.append(L_1[i][1], 1)$;
      else if $L_2[j-1][0] \equiv$ “e” and $L_2[j][0] \equiv$ “s” **then**
        $L_1[i][0] \equiv \text{“}s\text{”} \ ? \ L_3.append(L_1[i][1], 1) : L_3.append(L_1[i][1], 0)$;
    $i = i + 1$;
  else if $L_1[i][1] \equiv L_2[j][1]$ **then**
    if $i \equiv 0$ and $j \equiv 0$ **then**
      $L_3.append(L_1[i][1], 2)$;
    else if $i \equiv (2 \cdot l_1 - 1)$ and $j \equiv (2 \cdot l_2 - 1)$ **then**
      $L_3.append(L_1[i][1], 0)$;
    else
      $L_3.append(L_1[i][1], 1)$;
    $i = i + 1, j = j + 1$;
  else
    if $i \equiv 0$ and $j \equiv 0$ **then**
      $L_3.append((L_2[j][1], 1))$;
    else
      if $L_1[i-1][0] \equiv$ “s” and $L_1[i][0] \equiv$ “e” **then**
        $L_2[j][0] \equiv \text{“}s\text{”} \ ? \ L_3.append(L_2[j][1], 2) : L_3.append(L_2[j][1], 1)$;
      else if $L_1[i-1][0] \equiv$ “e” and $L_1[i][0] \equiv$ “s” **then**
        $L_2[j][0] \equiv \text{“}s\text{”} \ ? \ L_3.append(L_2[j][1], 1) : L_3.append(L_2[j][1], 0)$;
    $j = j + 1$;
while $i < 2 \cdot l_1$ **do**
  if $L_1[i][0] \equiv$ “s” **then**
    $L_3.append((L_1[i][1], 1))$;
  else
    $L_3.append((L_1[i][1], 0))$;
while $j < 2 \cdot l_2$ **do**
  if $L_2[j][0] \equiv$ “s” **then**
    $L_3.append((L_2[j][1], 1))$;
  else
    $L_3.append((L_2[j][1], 0))$;
$\Gamma_3.append(v_{ID})$;
for k in $[0, |L_3| - 2]$ **do**
  if $L_3[k][1] \equiv 0$ **then**
    $continue$;
  Initialize an empty list $temp$;
  $temp.append(L_3[k][0], L_3[k+1][0], L_3[k][1])$;
  $\Gamma_3.append(temp)$;
return Γ_3

preview vp for one specific video $v \in \mathsf{V}$, and other videos in V can be applied to the same functionality. Therefore, we simplify the playback records Γ_i for user i as $(v_{ID}, v, \langle t_s, t_e, c\rangle)$, namely considering one specific v instead of a sequence of

Algorithm 3 $Generate()$ - Video preview generation

Input: Γ', T_1, T_2 (the Γ' is the output of Algorithm 1)
Output: The video preview vp for a specific video v

Initialize an empty map $temp$, and an empty list vp;
$vp\_length = 0$;
$(v_{ID}, \langle t_s, t_e, c\rangle) \leftarrow \Gamma'$;
Sort items in $\langle t_s, t_e, c\rangle$ based on the descending order of $(t_e - t_s)$, and append to $temp$;
for i in $[0, |temp|]$ **do**
  $vp\_length = vp\_length + temp[i]$;
  if $vp\_length \leq T_1$ **then**
    $vp.append(temp[i])$;
  else if $T_1 \leq vp\_length \leq T_2$ **then**
    return $vp.append(temp[i])$
  else
    Keep the first $(T_2 - T_1)$ seconds in $temp[i]$ as $temp[i]'$;
    return $vp.append(temp[i]')$
return vp

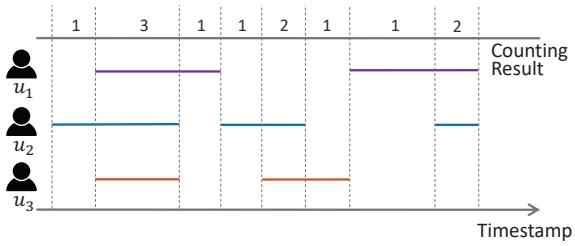

Fig. 2. An example of counting the union of playback segments from $n = 3$ users for a specific video v.

videos that have been replayed. We propose to instantiate the v_{ID} as the hash value of v, and the uniqueness of v_{ID} is guaranteed by the one-way, collision-resistant properties of the hash function. For the specific v, we consider there are n users who left playback records for it and therefore these $\Gamma_i, i \in [n]$ share the same v_{ID}, v in such a standalone setting. With this information, $\mathcal{F}$ operates as the following steps:

Step 1. counting. This step, i.e., the subroutine $Count$, aims to count the replayed times for the union of the playback video segments from n users. To ensure that users contribute fairly to the counting result, each user is only counted once no matter how many times he replayed for a certain video segment. An example is depicted in Figure 2. In the $Count$ subroutine, we utilize the *divide-and-conquer* paradigm to reduce the operation steps as the counting process for n users can be recursively broken down into multiple sub-procedures of counting for two users, which is a subroutine introduced in Algorithm 2, where a variant of *merge sort* is performed. Specifically, the number of playback video segments for the two users is denoted with l_1 and l_2 respectively. We consider that the segments are ordered and independent in the playback records in the sense that the $segment_{i-1}$ ($i \in [l_1]$ or $i \in [l_2]$) is always stored before (i.e., with lower index) $segment_i$ in the playback records Γ if $segment_{i-1}.t_e \leq segment_i.t_s$, and any two segments do not overlap with other. The output of this step is the sum of counting from all the n users for the same specific video v. Essentially, the result demonstrates the preference of each video segment (i.e., the playback popularity δ) in the output to all users who left playback records.

Authorized licensed use limited to: Miami University Libraries. Downloaded on June 14,2021 at 18:47:35 UTC from IEEE Xplore. Restrictions apply.

Step 2. generation. In this step, the subroutine $Generate$ (i.e., Algorithm 3) takes as input the accumulated counting result for v and outputs the target video preview vp. We omit the details of sorting the items in the counting results (i.e., Γ') as efficient sorting algorithms can be the candidate such as *quick sort*. After sorting, multiple segments can be selected and concatenated as the vp. The length of vp is constrained by T_1 and T_2. One possible unexpected situation is that the segment with the minimum duration, i.e., $(t_e - t_s)$, in vp may be longer than $T_2 - T_1$. In this case, we propose to keep the first $(T_2 - T_1)$ seconds in the segment with minimum duration in vp to ensure the length of vp being in the expected range, i.e., $T_1 \leq |vp| \leq T_2$. A more appealing while complex way to generate vp is to extract several sub-segments with key-frames from the counting result, and we leave this for the future work.

It is worthy to point out that for simplicity, Algorithms 1, 2, and 3 are employed to generate the video preview for one specific video $v \in \mathsf{V}$. The functionality $\mathcal{F}$, however, can repeat these algorithms/procedures to generate previews for all other videos in V. Overall, the $\mathcal{F}$ achieves the functionality of taking as input the video set V and outputs the video preview set VP.

Lemma 1. *The time complexity of the functionality $\mathcal{F}$ is optimal.*

Proof. The complexity for the $Merge$ subroutine is $O(l_1 + l_2)$, where l_1 and l_2 represent the number of playback video segments for two users, and therefore the complexity for the $Count$ process is $O(n \cdot (l_1 + l_2))$, which is theoretically optimal. Meanwhile, the $Generate$ subroutine utilizes a *greedy* algorithm to find the video preview vp from the sorted playback video segments. The problem of finding the vp is equivalent to the continuous knapsack problem, which can be solved in polynomial time, i.e., $O(n \log(n))$, where n is the number of playback segments in the counting result and the main bottleneck is caused by the sorting process, and the greedy solution is optimal [22]. Conditioned on that the two subroutines of $\mathcal{F}$ is optimal and the fact that the two subroutines are independent with each other, the complexity of $\mathcal{F}$ is optimal.

Lemma 2. *The functionality $\mathcal{F}$ finds the video preview with the maximum playback popularity Δ for v.*

Proof. Each playback video segment in the counting result is sorted by the total number of replayed times from all users who left playback records. The greedy solution in $Generate$ ensures that the vp is the concatenation of the segments with the highest playback popularity δ. Consequently, the video preview popularity of Δ is maximum.

V. Experiment and Analysis

To evaluate whether the video preview could be more attractive to the viewers, we recruit 120 young student volunteers (we would use the "volunteers" and "viewers" interchangeably) including 60 females and 60 males aging from 20 to 26 years old in the university, and randomly divide them into two groups, each of which includes 30 males and 30 females.

Program ID	Other Metadata	Segment_1_playback_records	Segment_2_playback_records	Other Segments
36615041	…	[‘2019-11-04 20:13:02’, ‘2019-11-04 20:13:06’, 186]	[‘2019-11-06 10:08:27’, ‘2019-11-06 10:08:30’, 273]	…
36686395	…	[‘2019-11-07 21:28:53’, ‘2019-11-07 21:28:55’, 406]	[‘2019-11-08 14:36:04’, ‘2019-11-08 14:36:07’, 384]	…
36706731	…	[‘2019-11-16 09:37:56’, ‘2019-11-16 09:38:01’, 322]	[‘2019-11-17 19:41:32’, ‘2019-11-17 19:41:34’, 315]	…

Fig. 3. An exemplary dataset format of the counting result during the procedure of generating video preview for videos/programs.

Meanwhile, we cooperate with a digital television media company[2] whose main business contains the dissemination of visual and textual television programs on a subscription or fee basis, and obtain playback records of 3,706 programs/videos over 262,838 users. We remove the playback records for the video segments that less than 30 seconds and eventually garner 392,837 valid records. In our experiment, we randomly select 15 videos with 1,790 playback records covering 1,063 users and generate the video preview for them using the proposed algorithms. Figure 3 presents an exemplary data format of the counting result during the procedure of generating video preview. For each program, we also generate a static thumbnail image, and both the preview and the thumbnail would link to the corresponding full-length program. Then the volunteers in the two groups are assigned with content in the following way:

Content-factored group. For the first group of viewers, we display both the thumbnail image and the video preview in an adjacent way for each of the 15 programs, leading to 30 items in the web interface. Then the number of clicks for different programs is collected. This group of evaluation is content-factored in the sense that viewers are easy to find both the preview and thumbnail for the same content. By comparing the number of clicks for the same program from different viewers, we could clearly perceive the preference for these two different displaying styles.

Position-factored group. For the second group of viewers, we also create both the video preview and the static thumbnail image for the 15 programs, and therefore, there are 30 items shown on the web page. However, the location of the 30 items is totally random, namely, they are randomly ordered and displayed. Again, we collect the total clicks for each program. As the viewers' click on an item may not only because (s)he favors but also because it is in a good position [23]. This group of evaluation is position-factored in the sense that we would like to mitigate the impact factor of position and compare the preference of different displaying styles.

Experimental result analysis. Figure 4 and 5 depict the experimental results from the two groups and the concrete results are tabulated in Table II. We thus have the following observations: In both the content-factored and position-factored groups, the number of clicks for the video preview exceeds the number of clicks for the static thumbnail images.

VI. Conclusion

In this paper, we propose a method to produce video preview from users' playback records. The key idea is to identify users'

[2]https://www.wasu.com.cn/

Authorized licensed use limited to: Miami University Libraries. Downloaded on June 14,2021 at 18:47:35 UTC from IEEE Xplore. Restrictions apply.

TABLE II
THE CLICK TIMES FOR TWO DIFFERENT GROUPS.

Program type & Program(Video) index		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Content-factored group	Static Thumbnail Image	33	21	16	31	28	19	17	34	15	24	20	27	32	33	23
	(Short) video preview	37	21	20	29	33	36	36	33	38	39	24	30	36	26	29
Position-factored group	Static thumbnail image	30	26	24	19	19	18	16	13	12	11	11	10	10	10	10
	(Short) video preview	31	30	29	26	25	24	24	24	22	21	20	20	20	20	20

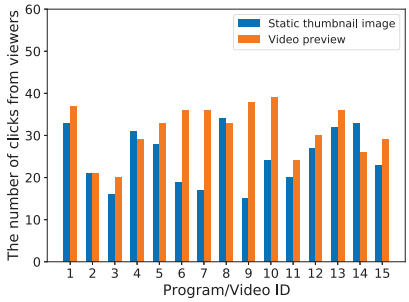

Fig. 4. The comparison of content-factored group.

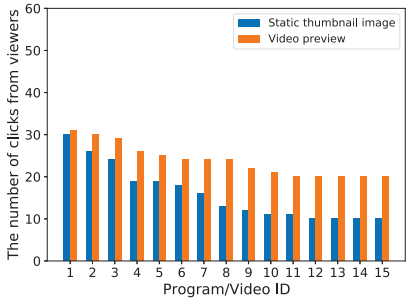

Fig. 5. The comparison of position-factored group.

common preferences to generate the most popular segments as video preview for other users. We presented the details of algorithm design and conducted experiments to demonstrate the feasibility. Our work is complementary to recommendation systems that recommend the most relevant programs/videos to users based on their personal preferences. Our future work lies in the design of progressive video preview generation and consider more factors for performance enhancement.

ACKNOWLEDGMENT

This work was supported by the Key Research and Development Program of Zhejiang Province, China under Grant No. 2019C03138 with the Communication University of Zhejiang, China.

REFERENCES

[1] Cisco, "Cisco Annual Internet Report (2018-20230) White Paper," https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html, 2020.
[2] TikTok, https://www.tiktok.com/, 2020.
[3] T. Information, "YouTube Plans 'Shorts' to Rival TikTok," https://www.theinformation.com/articles/youtube-plans-shorts-to-rival-tiktok, 2020.
[4] P. Leskin, "Take a closer look at Instagram Reels, Facebook's TikTok rival launching today in the US," https://www.businessinsider.com/instagram-reels-tiktok-competitor-short-video-us-launch-explainer-2020-7, 2020.
[5] T. Star, "Tencent-backed Kuaishou launches short video app for global audience – and it looks similar to TikTok," https://www.thestar.com.my/tech/tech-news/2020/04/28/tencent-backed-kuaishou-launches-short-video-app-for-global-audience—and-it-looks-similar-to-tiktok, 2020.
[6] B. Singh, M. Folgner, R. Cunningham, D. Regan, Y. Wang, N. Vihinen, and T. S. Woolway, "Video preview generation," Jul. 25 2017, US Patent 9,715,901.
[7] Y.-F. Ma, B. Lin, Z. Kong, X. Zou, W.-Y. Ma, and H.-J. Zhang, "Systems and methods for smart media content thumbnail extraction," Jul. 26 2011, US Patent 7,986,372.
[8] S. A. Esmaeili, B. Singh, and L. S. Davis, "Fast-at: Fast automatic thumbnail generation using deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4622–4630.
[9] P. D. Gunatilake, "Video preview module to enhance online video experience," May 7 2013, US Patent 8,438,484.
[10] D. McIntosh and C. Pennello, "Video preview creation based on environment," Aug. 7 2014, US Patent App. 14/173,732.
[11] C.-H. Lin and H. Chi, "A novel movie recommendation system based on collaborative filtering and neural networks," in *International Conference on Advanced Information Networking and Applications*. Springer, 2019, pp. 895–903.
[12] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 191–198.
[13] M. Chen, A. Beutel, P. Covington, S. Jain, F. Belletti, and E. H. Chi, "Top-k off-policy correction for a reinforce recommender system," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 456–464.
[14] X. Chen, Y. Zhang, Q. Ai, H. Xu, J. Yan, and Z. Qin, "Personalized key frame recommendation," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017, pp. 315–324.
[15] W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, and Z. Qin, "Aesthetic-based clothing recommendation," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 649–658.
[16] B. S. Neysiani, N. Soltani, R. Mofidi, and M. H. Nadimi-Shahraki, "Improve performance of association rule-based collaborative filtering recommendation systems using genetic algorithm," *I.J. Information Technology and Computer Science*, pp. 48–55, 2019.
[17] J. Gao, T. Zhang, and C. Xu, "A unified personalized video recommendation via dynamic recurrent neural networks," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 127–135.
[18] M. Redi, N. O'Hare, R. Schifanella, M. Trevisiol, and A. Jaimes, "6 seconds of sound and vision: Creativity in micro-videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4272–4279.
[19] L. Huang and B. Luo, "Personalized micro-video recommendation via hierarchical user interest modeling," in *Pacific Rim Conference on Multimedia*. Springer, 2017, pp. 564–574.
[20] J. Chen, X. Song, L. Nie, X. Wang, H. Zhang, and T.-S. Chua, "Micro tells macro: Predicting the popularity of micro-videos via a transductive model," in *Proceedings of the 24th ACM international conference on Multimedia*, 2016, pp. 898–907.
[21] Wikipedia, "Continuous knapsack problem," 2020.
[22] K. Bernhard and Y. Jens, "Fractional knapsack and weighted median," in *Combinatorial Optimization: Theory and Algorithms, Algorithms and Combinatorics*. Springer, 2012, vol. 21, pp. 459–461.
[23] H. Guo, J. Yu, Q. Liu, R. Tang, and Y. Zhang, "Pal: a position-bias aware learning framework for ctr prediction in live recommender systems," in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, pp. 452–456.

Authorized licensed use limited to: Miami University Libraries. Downloaded on June 14,2021 at 18:47:35 UTC from IEEE Xplore. Restrictions apply.