AutoTransition: Learning to Recommend Video Transition Effects

Yaojie Shen[1,2,3], Libo Zhang[1,2], Kai Xu[3], and Xiaojie Jin[3(✉)]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] ByteDance Inc., Beijing, China
jinxiaojie@bytedance.com

Abstract. Video transition effects are widely used in video editing to connect shots for creating cohesive and visually appealing videos. However, it is challenging for non-professionals to choose best transitions due to the lack of cinematographic knowledge and design skills. In this paper, we present the premier work on performing automatic video transitions recommendation (VTR): given a sequence of raw video shots and companion audio, recommend video transitions for each pair of neighboring shots. To solve this task, we collect a large-scale video transition dataset using publicly available video templates on editing softwares. Then we formulate VTR as a multi-modal retrieval problem from vision/audio to video transitions and propose a novel multi-modal matching framework which consists of two parts. First we learn the embedding of video transitions through a video transition classification task. Then we propose a model to learn the matching correspondence from vision/audio inputs to video transitions. Specifically, the proposed model employs a multi-modal transformer to fuse vision and audio information, as well as capture the context cues in sequential transition outputs. Through both quantitative and qualitative experiments, we clearly demonstrate the effectiveness of our method. Notably, in the comprehensive user study, our method receives comparable scores compared with professional editors while improving the video editing efficiency by **300** $\times$. We hope our work serves to inspire other researchers to work on this new task. The dataset and codes are public at https://github.com/acherstyx/AutoTransition.

Keywords: Video transition effects recommendation · Multi-modal retrieval · Video editing

Y. Shen, L. Zhang, K. Xu and X. Jin—Equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-19839-7_17.

S. Avidan et al. (Eds.): ECCV 2022, LNCS 13698, pp. 285–300, 2022.
https://doi.org/10.1007/978-3-031-19839-7_17

1 Introduction

With the advance of multimedia technology and network infrastructures, video is ubiquitous, occurring in numerous everyday activities such as education, entertainment, surveillance, etc. There is a massive amount of needs for people to edit videos and share with others. However, video editing is challenging for non-professionals since it is not only laborious but also needs a lot of cinematography and design knowledge. Some editing tools like *Adobe Premier* and *Apple Final Cut Pro* are developed to assist video editing, however their main target users are professionals while novices may find it difficult to learn. Moreover, they still lack the ability of automatic video editing, i.e., users have to manipulate videos on their own. Recently, popular video editing tools like *InShot Video Editor* and *CapCut* provide the function of creating videos in one-click. Nevertheless, since they only utilize simple strategies or fixed video templates and ignore the content of input vision/audio, the quality of generated video is unsatisfactory.

The video transition effects play an important role in video editing to join shots for creating smooth and cohesive videos. In this paper, we introduce a new task of automatic video transition recommendation (VTR) and provide a systematic solution. Specifically, VTR is defined as: given a sequence of raw video shots and the companion audio (which can either be original sound or overwritten music), recommend a sequence of video transitions for each neighboring shot. Different from conventional classification problems which choose only one most probable category as the output, VTR aims to provide a ranking of candidate transition categories so that users can choose freely in practical usage.

When working on this task, we encounter following challenges. First, there is no publicly available video transition dataset for training and evaluation. It takes enormous efforts to manually collect and annotate a large-scale video dataset. Meanwhile, due to the large complexity and diversity of video editing, the evaluation of video quality is subjective and vary from person to person. Thus during creating the dataset, it is crucial to design proper criteria for selecting video samples that are appealing to most. Besides dataset, solving VTR is also challenging. A good video transitions recommender should ensure top transitions match well with both the dynamics and contents of videos and the rhythm and theme of audio (or music). Moreover, transitions recommended at multiple video connections should be harmonious so that the final video is visually smooth and unified. Being the first effort in addressing VTR, we need to take all of above factors into consideration for delivering the optimal solution.

We start with building a video transition dataset from those video editing templates that are publicly available on video editing tools. We design comprehensive rules via trials to select high-quality video templates followed by pre-processing for refinement. Afterwards, we extract video shots and corresponding transitions (used as ground-truth in training), creating the first large-scale video transitions dataset. More details are introduced in Sect. 3.

To figure out the best way for modeling VTR, we conduct extensive experiments to compare the classification-based and retrieval-based solutions and finally demonstrate the latter performs better. In the classification-based solution, the model takes neighboring video shots as inputs and outputs the prediction of transition categories. The cross-entropy loss between predictions and ground-truth transitions is used. In the retrieval-based solution, we first pre-train a transition classification network to learn the embedding of transitions. Then we propose a multi-modal transformer to learn the fused vision/audio features in sequential video shots. A triplet margin loss is devised to minimize the distance between fused input features and pre-trained transition embedding. Similarly, in testing, according to the distance calculated between input features and transition embedding, the transition categories with smaller distances are in higher ranking position.

To conclude, our contributions are threefold:

1. We introduce a new task of automatically recommending video transition effects given video and audio inputs. We collect the first large-scale video transitions dataset to facilitate future research on this task.
2. We formulate the transition recommendation task as a multi-modal retrieval problem and propose a framework for learning the correspondence between input vision/audio and video transitions in feature space. The proposed framework is capable of fully utilizing the multi-modal input information for generating sequential transition outputs.
3. Through both quantitative and qualitative evaluation, we demonstrate that the proposed method can successfully learn the matching from vision/audio to transitions and generate reasonable recommendation results. Moreover, a user study conducted to evaluate the quality of generated videos further demonstrates the effectiveness of our method.

2 Related Work

Video Editing. Automatic video editing is challenging due to following reasons. First, the model has to fully understand the spatial-temporal context and multi-modal information in videos to obtain semantically coherent results. Second, video editing requires lots of professional knowledge to endow videos with creativity and particular aesthetic taste. Third, the evaluation of the quality of the generated video may be subjective. Recently, there are some progresses towards performing automatic video editing, each focusing on different aspects. Frey et al. [1] proposes an automatic approach to transfer the editing styles of an edited video to the new raw shots. Wang et al. [2] builds a tool for creating video montage based on the text description. Koorathota et al. [3] proposes a method to perform video editing according to a short text query by utilizing contextual and multi-modal information. Liao et al. [4] introduces a method for music-driven video montage. Several methods focus on solving video ordering and shot selection [1,2,5]. Distinguished from all tasks above, VTR is still unexplored although its equal importance in creating high-quality videos in video editing. In this work, we take the first step to close the research gap.

Video Transitions. Video transition is a widely used post-production technique for achieving smooth transitions between neighboring shots via special image/video transforms. There are various kinds of transitions including straight cuts, fades, and 3D animations among many others. To professionals, each type of transition is with a dedicated meaning to convey specific emotions, feelings or scene information to viewers, thus should be used meticulously. Moreover, when multiple video transitions are used, they should work in a harmonious way to ensure the visual unification of the final video. Due to above reasons, it is difficult for non-professionals to apply video transitions in their edit. Our work can substantially assist these people by automatically recommending reasonable video transitions on the fly.

Visual-semantic Embedding. Many recent works on video retrieval are based on the alignment of visual-semantic embedding [6,7]. Embedding techniques are employed to measure the similarity of different modalities in cross-modal video retrieval tasks, where features from different modals are mapped into a shared embedding space for better alignment [8,9]. Miech et al. [9] utilizes millions of video text pairs to learn the text-video embedding. Escorcia et al. [10] aligns the embedding of text and moments in the videos. They share the same idea of jointly aligning embeddings from two different modals. Triplet loss is initially proposed for learning the distance metric [11]. It is used for learning multi-modal embedding through deep neural networks in recent works [12,13]. By optimizing the distance directly with a soft margin, triplet loss is suitable for ranking tasks [14]. Our method also employs triplet loss to learn the distance between representations. A multi-modal transformer is used therein to learn the features of vision/audio inputs which aim to match with the pre-trained video transition embedding. Through extensive experiments, we demonstrate the effectiveness of our methods.

Multi-modal Transformer and Sequence Modeling. Our task is closely related to recent progress in modeling the spatial-temporal and multi-modal information in vision, speech and text. Transformer [15] is widely used in these tasks to encode cross-modal and spatial-temporal information [16–19]. It employs an attention mechanism to represent multi-modal information in a common latent space. In other sequential problems, Lin et al. [20] uses a modality-specific classifier and a differentiable tokenization scheme to fuse multi-modal information via transformer. Gabeur et al. [17] introduces a video encoder architecture with multi-modal transformer for video retrieval. We also use modal-specific networks to extract embeddings from vision and audio inputs. Specifically, we use SlowFast [21] and Harmonic CNN [22] to extract video and audio features respectively. Different with the vanilla transformer architecture which adopts an encoder-decoder architecture [15], recent works [23] proposes unified encoder-decoder transformer model for sequential modeling. In our method, we exploit a multi-modal transformer for learning the multi-modal representations as well as capturing the context information in sequential transitions.

3 Task Definition

We aim to solve the task of video transitions effect recommendation (VTR), the goal of which is to recommend appropriate transitions between neighboring video shots. As shown in Fig. 1a, we take a sequence of raw video shots $\{v_1, v_2, \ldots, v_n\}$ as inputs. To simplify the task, we assume that the order of the videos is already determined, all the videos are already cut and scaled to the target range, and the background audio (either original sound and/or overwritten music) is already specified. Then for a pair of video shots v_k and v_{k+1}, a transition effect is added to join them. We denote the video clip added with transition effect as $t_{k,k+1}$. Since video transitions generally mix neighboring video shots, we separate the final video after adding transitions into two parts, one is the clips added by transitions (i.e. $t_{k,k+1}$), the other is uncontaminated video shots (i.e. v'_k and v'_{k+1}). Then the output video after adding transitions can be denoted as $v' = \{v'_1, t_{1,2}, v'_2, \ldots, t_{n-1,n}, v'_n\}$. The dataset we collect contains both the start and end timestamps of each transition so that we can obtain the exact positions of video shots and the categories of transitions. Note that in the collected dataset, we can only get access to the output video v' since the original videos v are not publicly available.

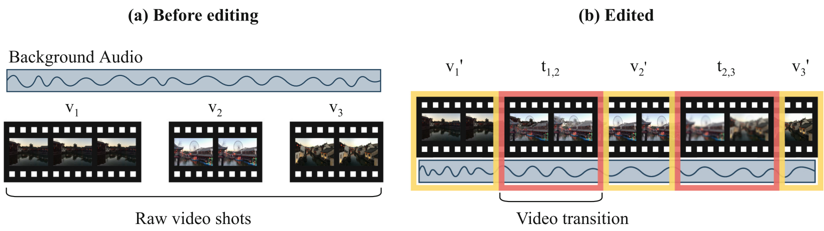


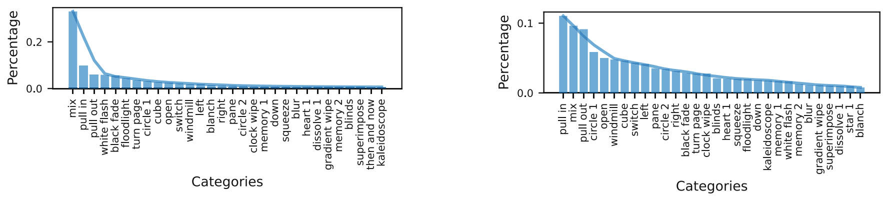
Fig. 1. The definition of the task. We take a sequence of raw video shots, for example $\{v_1, v_2, v_3\}$, and companion audio as the inputs, the task is to recommend categories of the transitions $\{t_{1,2}, t_{2,3}\}$ used in the edited video.

In our method, we use the video clip $t_{k,k+1}$ and the label of corresponding transition effect $c_{k,k+1}$ to train the transition classification network, for learning the embedding of transitions based on their visual representation. When training the transition recommendation model, we remove $t_{k,k+1}$ from the input to be consistent with the inference setting where input videos are without any transitions. The uncontaminated video shots v'_k and v'_{k+1} are used to represent the original video shots, i.e. v_k and v_{k+1} respectively. This strategy is reasonable since the duration of transition effect is short, v'_k and v'_{k+1} can serve as a good approximation to their original counterparts.

4 Video Transition Dataset

4.1 Raw Data Collection

With the development of video editing tools and platforms, a large amount of well-designed video templates are publicly available. Produced by professionals, video templates define fixed combinations of essential editing elements, including the number of video shots, the length of video, music, transitions, animations, and camera movement, etc. By simply replacing materials in templates with self videos, even novice users can create edited videos easily. Each video template also comes with an example video made by the designer using this template and his/her original videos. In our experiments, we collect video templates from these online video platforms and get the annotations related to transition effects, including each transition's category and corresponding start/end time. Though video templates may contain other special visual effects like animation and 3D movements, we only consider video transition effects in this paper.



(a) The label distribution on the small dataset at the first stage of data collection.

(b) The label distribution on the final dataset after filtering.

Fig. 2. Label distribution of top-30 categories in the dataset. (a) The label distribution on the small dataset at the first stage of data collection. (b) The label distribution on the final dataset after filtering.

Table 1. The statistical information of the transition dataset.

Dataset	Train set	Test set
Number of videos	29,998	5,000
Number of transitions	118,984	19,869
Transitions per video	3.966	3.973
Average video length	15.83 s	15.79 s

4.2 Data Filtering

We perform data filtering to improve the quality and diversity of the dataset. At first, we limit the maximum length of the video to 60 s, and the templates are filtered depending on the number of user likes and usage. We ignore those videos without any transition. We gather a small dataset through manual crawling and examine the overall distribution of collected samples. To guarantee that there are enough training samples for each category, we only select top-30 categories for training and testing according to the amount of samples. By statistical results, we observe that there are many duplicated transitions in a single video, and different types of transitions are distributed in a severe long-tail manner. The label distribution of the small dataset is shown in Fig. 2a.

We believe that the duplication and long-tailed distribution are harmful to the diversity of recommendation. To solve this issue, we use two additional rules to select samples: each video should contain more than two different types of transitions, and the usage times of the same transition should be no more than six. Following these rules, we acquire the final dataset which contains 34998 videos (train-29998 and test-5000) in total and 138.8K valid transitions between neighboring video shots. Table 1 shows more statistical results of the dataset. The label distribution is shown in Fig. 2b.

5 Video Transition Recommendation

5.1 Pre-training Transition Embedding

We formulate the video transition effects recommendation as a multi-modal retrieval problem. Since retrieving from vision/audio to video transitions requires the model to learn correspondence between vision/audio inputs and video transitions, the first problem we need to solve is thus how to learn a strong representation for each transition. We notice that some video transitions have similar visual effects like move up and move down. It is natural that we expect the learned embedding can also reflect these connections. To achieve this goal, we employ a video classification network to learn transition embedding based on their visual appearance. As shown in Fig. 3, we take the transition clips t as input and use the video backbone to extract visual representations. After passing through linear transform and normalization, we obtain a unit vector for each transition. Then we apply another linear transform and use the cross-entropy loss to optimize the classification objective. As expected, the embedding of the transitions are separably distributed in latent space and similar transitions stay close to each other. The visualization result of learned embeddings through t-SNE is illustrated in Fig. 5.

5.2 Multi-modal Transformer

As shown in Fig. 4, we propose a multi-modal transformer to extract representations from the raw video shots and audio. For recommending a transition $t_{k,k+1}$

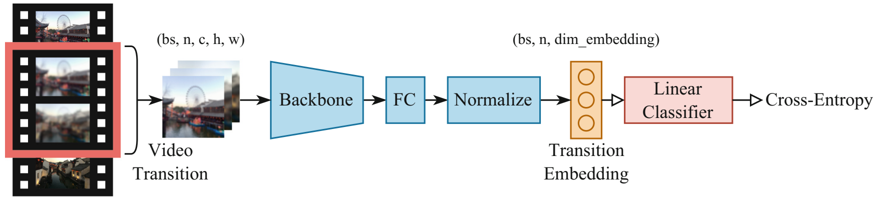


Fig. 3. Extracting transition embedding. A transition classification network is built to learn the transition embedding.

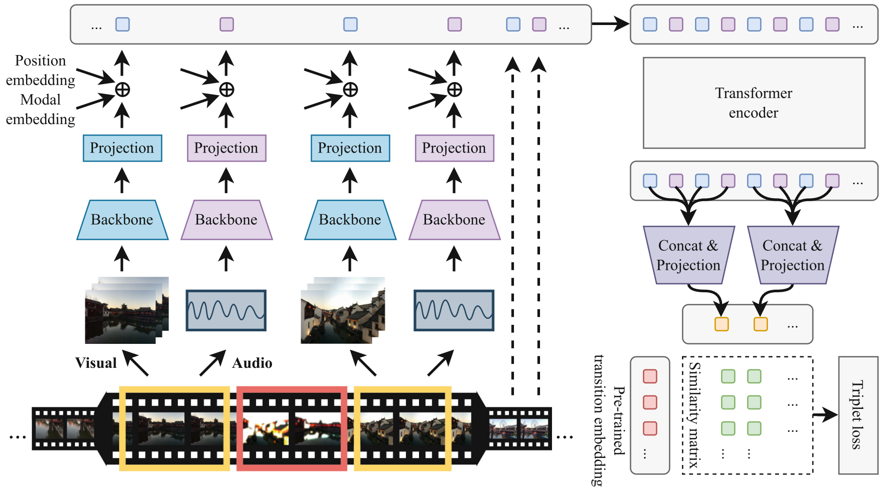


Fig. 4. Transition recommendation model for retrieving matching transitions based on vision/audio inputs. First, we use modality-specific networks to extract visual and audio features. After that, a multi-modal transformer encoder fuses the tokens from different modalities. Finally, a triplet loss is used to optimize the network end-to-end.

between video shots v_k and v_{k+1}, we take both the video frames and audio waves from the end period of video shot v_k and the start period of video shot v_{k+1} as input. Specifically, n video frames are sampled uniformly from each video shot.

After obtaining video frames and corresponding audio waves, we extract their features by feeding into visual and audio backbone respectively. Note that these backbones can be conveniently replaced by other common video or audio models. In our experiments, we use the SlowFast [21] and the Harmonic CNN [22] as video and audio backbone respectively.

In order to make full use of multi-modal information of vision/audio, we combine the visual and audio features as the input for the multi-modal transformer. By doing so, the model not only learns the matching relationship from input to transitions but also captures the context cues among sequential transitions.

Before being fed into the transformer, visual tokens and audio tokens are projected to the same dimension by independent linear transformations. Then learnable positional embedding and modal embedding are element-wisely added to these tokens. We share the positional embedding for vision/audio tokens at the same time point. Modal embedding is applied to inform the model which modality the token belongs to. After above processing, tokens from all video shots are input into transformer as a whole. In this way, the transformer is encouraged to learn the contextual relationship in sequential transitions. As demonstrated by experimental results, such a context-aware training method contributes to generate harmonious sequential outputs. The self-attention mechanism in transformer encoder can model complex mutual relationship among input tokens. From all output tokens, we concatenate those four tokens which corresponds to each transition point (for each modality of vision and audio, there are two tokens before and after the transition point) along the feature dimension and get the final representation by a projection to the same dimension of transition embedding.

5.3 Transition Recommendation

As introduced in Sect. 5.2, the multi-modal video embedding is extracted by the multi-modal transformer, which is used as the query to retrieve the transitions by matching the pre-trained transition embedding mentioned in Sect. 5.1. During retrieval, we apply learnable linear transformations to both the pre-trained transition embedding and the video embedding for achieving better alignment.

We denote $E^{trans} = \{e_1^{trans}, \ldots, e_N^{trans}\}$ as the set of pre-trained transition embedding where N is the number of categories and $N = 30$ in our experiments. $E^{video} = \{e_1^{video}, \ldots, e_V^{video}\}$ is denoted as the multi-modal input embedding in a batch. V is the number of all transition points in a batch. For each sample e_v^{video}, we calculate its matching score with every transition embedding through a similarity metric $\Phi(e_v^{video}, e_k^{trans})$. In our implementation, Φ adopts the form of dot-production due to its simplicity, i.e.

$$\Phi(e_v^{video}, e_k^{trans}) = \langle e_v^{video}, e_k^{trans} \rangle. \tag{1}$$

Eq. (1) is used to calculate the ranking loss in the following training steps.

Training. We expect that the model can learn to rank transitions by their distances with inputs in embedding space. To achieve so, we utilize the triplet margin loss to optimize the similarity between the transition embedding and multi-modal video embedding. For the embedding of each training sample e^{video} with ground truth label c, we define our training objective with triplet loss as

$$\mathcal{L}(e^{video}) = \frac{1}{N-1} \sum_{k \neq c, k \in \{1, \ldots, N\}} \mathcal{T}(e^{video}, e_c^{trans}, e_k^{trans})$$

where $\mathcal{T}$ calculates the triplet margin loss for each triplet $(e^{video}, e_c^{trans}, e_k^{trans})$:

$$\mathcal{T}(a, p, n) = max(\Phi(a, p) - \Phi(a, n) + M, 0).$$

M is the soft margin, a, p and, n are anchor, positive sample, and negative sample, respectively. Φ follows the definition of Eq. (1). In our settings, we take video embedding e^{video} as the anchor, the transition embedding with category c as the positive sample, others transitions as negative samples. The final objective is the average loss over all samples, i.e.

$$\mathcal{L}(E^{video}) = \frac{1}{V} \sum_{v \in \{1,\ldots,V\}} \mathcal{L}(e_v^{video}). \tag{2}$$

By optimizing Eq. (2), the model encourages the similarity between the multi-modal video embedding and its ground-truth transition embedding higher than the similarity between non-matching pairs with a margin of M.

Evaluation. In evaluation, we follow Eq. (1) to measure the matching degree between multi-modal video embedding and candidate transitions embedding. For two neighboring video shots, we sort the similarities of candidate transitions in descending order and select the top one as final result.

6 Experiments

6.1 Implementation Details

Model Details. We employ SlowFast 8×8 [21] as the backbone to extract visual features. The same backbone is also used as the transition embedding network. By default, we freeze SlowFast from stage 1 to stage 3 during our experiments to save memory. We train all models on one machine with 8 NVIDIA V100 GPUs, except the experiment without freezing the SlowFast backbone, in which we use two machines and 16 GPUs in total. For audio modal, we use Harmonic CNN [22] to extract the local audio features around the transition. Then we linearly project the feature of two modals to the same dimension of d_{model} and take them as the input tokens of the multi-modal transformer. The multi-modal transformer consists of two transformer layers with $d_{model} = 2048$ and $n_{head} = 8$. Before matching, we apply linear projections to both the video embedding and pre-trained transition embedding in order to map them into a joint space. Such a projection is experimentally proved to be beneficial as shown in Table 3.

Data Preprocessing and Training. When training the transition embedding network, we uniformly sample 16 frames with the image size of 224×224 from the transition duration as the input. The batch size is set to 256. The training process is 30 epochs in total, and start with a warm up for 5 epoch to raise the learning rate from 1e-6 to the initial learning rate 1e-3, then decay by a factor of 0.1 every 10 epochs. We use the model parameters of the last epoch to generate the transition embedding.

When training the transition recommendation model, we uniformly sample 16 frames with the image size of 224×224 as the visual inputs. The local audio features are extracted using the pre-trained Harmonic CNN before training. Given a time point, the Harmonic CNN generate a feature vector with a dimension of 100 based on the audio in around one second. For the sequential inputs, we set the maximum sequence length to 8. Redundant transitions beyond the maximum length are dropped while zero tensors are padded if the length is less than 8. We use Adam optimizer in all the experiments and set the soft margin $M = 0.3$ in triplet margin loss. The initial learning rate is set to 1e-5, then decay by a factor of 0.1 every 10 epochs. The total training epoch is 30 epochs.

Metrics. For testing the fine-grained performance of our model, we evaluate the individual recommendation results in testing. The commonly used Recall@K metric ($k \in \{1, 5\}$) and Mean Rank are employed as evaluation metrics. Since in our dataset, there is only one ground-truth for each transition, Recall@K indicates the likelihood of hitting the target in top K retrieval results. Mean rank is the averaged rank of all ground-truth transitions, whose math formula is

$$\mathrm{MR} = \frac{1}{|\mathcal{S}|} \sum_{i \in S} \mathrm{Rank}(i)$$

where $\mathcal{S}$ represents the set of all transitions in test.

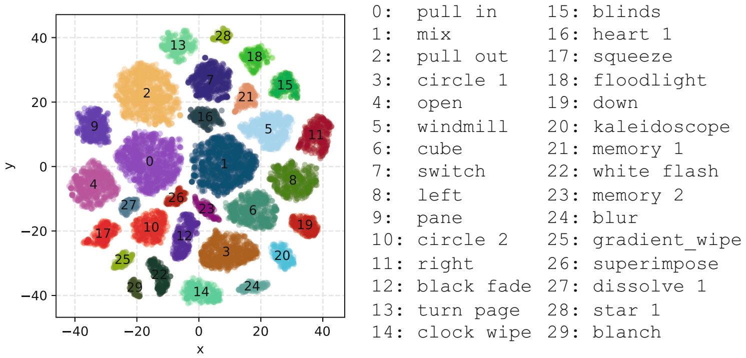


Fig. 5. t-SNE visualization of the transition embedding on the train set after pre-training. We use cosine similarity as the distance metrics when running t-SNE. We drop the outliers and randomly pick 10K samples from the t-SNE output for visualization. We can see that transitions with similar visual effects, such as “left” (8), “right” (11), and “down” (19) are close to each other in the embedding space.

6.2 Extracting Transition Embedding

A transition classification network is trained using annotated transition category in the dataset as the transition embedding network. After training, we use the pre-trained network to extract the embeddings of all transitions in the training set. Normalization is then applied for the embedding as shown in Fig. 3 to convert the embedding into a unit vector. We drop the outliers utilizing the three-sigma rule by assuming the embeddings follow a normal distribution. The embeddings after dropping are visualized in Fig. 5. The remaining embeddings for each category are averaged to generate the final transition embedding. We then demonstrate the effectiveness of the transition embedding by the experiments shown in Table 3. Notably, the transition embedding network is advantageous in extending the model to support new transition categories since retraining the recommendation model is circumvented.

6.3 Ablation Studies and Comparisons

We start by showing the advantages of leveraging contextual and multi-modal information in inputs. Then we verify the effectiveness of the pre-trained transition embedding by comparing it with random initialization. We also compare with the classification method to demonstrate the superiority of retrieval methods on this task. Due to space limit, more details of results are referred to the supplementary material.

Context and Multi-modal. In this experiment, we study the impact of contextual and multi-modal inputs on the recommendation performance, and the result is shown in Table 2. From the first and third rows of Table 2, we can see that sequential inputs introduce the context information to the model, which is helpful for modeling the temporal relations between the transitions. From the second and third rows of Table 2, visual modal inputs perform much better than audio as input, which indicates that visual content is more related to the transition effects than audio. The results in the last three rows demonstrate that the multi-modal inputs can improve the accuracy of recommendations than the single modal inputs.

Table 2. The impact of contextual and multi-modal information to the recommendation results.

Sequential (Context)	Modal		Recall@1	Recall@5	Mean Rank
	Visual	Audio			
	✓		24.12%	66.25%	5.758
✓		✓	19.39%	56.61%	7.012
✓	✓		25.40%	66.33%	5.665
✓	✓	✓	**28.06%**	**66.85%**	**5.480**

Table 3. Advantages of pre-training transition embedding from the visual effects. We demonstrate its effectiveness by comparing it with a random initialized embedding.

Transition Embedding	Projection	Recall@1	Recall@5	Mean Rank
Random initialization		25.67%	66.3%	5.646
Pre-trained transition embedding		26.24%	66.03%	5.623
Pre-trained transition embedding	✓	**28.06%**	**66.85%**	**5.480**

The Effectiveness of Pre-trained Transition Embedding. In this experiment, we study the effectiveness of our proposed pre-trained transition embedding, and the results are shown in Table 3. All the embedding is frozen during training. In the random initialization setting, we use a normalized random embedding as the replacement of the pre-trained transition embeddings. It is observed that the performance of using random embedding is worse compared to our pre-trained embedding. From the results in second and third rows in Table 3, the importance of the linear projection can be demonstrated. We conjecture the linear projection helps learning better mapping between the pre-trained transition embedding and multi-modal input embedding in a shared space (Table 4).

Table 4. Comparing with the classification method.

Methods	Recall@1	Recall@5	Mean Rank
Classification	22.27%	61.82%	6.099
Matching with pre-trained transition embedding	**28.06%**	**66.85%**	**5.480**

Comparing with the Classification Method. In this ablation study, we remove the transition embedding, replace triplet margin loss with cross-entropy, and train the recommendation model utilizing the transition category label. The comparison result is shown in Fig. 4. The classification model performs worse compared with the retrieval model. The reason is that the learned transition embeddings in retrieval model contain richer visual properties of the transitions compared with semantically meaningless one-hot vectors used in the classification model. In addition, the cross-entropy loss may impose excessive punishment for negative categories due to using one-hot ground-truth, thus neglecting the fact that there are similar transitions as the ground-truth transition and they can also be used as favorable alternatives.

6.4 User Study

Since the transition recommendation is subjective, the viewer's feeling is essential to the evaluation. Therefore, we conduct a user study to further verify its effectiveness. Specifically, we collect raw video shots from online copyright free video sources, e.g. videvo.net[1], covering various topics such as travel, life, entertainment, sports, nature, and animals. For each set of video shots, we fix their orders and assign an appropriate background music, leaving only the transitions between neighboring video shots to be added. After selecting video transitions, we use the tool of *CapCut* to connect the raw video shots by transitions, producing the final videos. We compare among following three methods of selecting video transitions.

1. **Weighted random pick.** At each transition point, select the transition category by a random sampling from a multinomial distribution. The probability of each category is its frequency in our collected video transition dataset.
2. **Professional video editor.** We ask a professional editor who has 6 years of video editing to select transitions. He is free to take as long as he wants to select the best transitions depending on his understanding of the given video/audio.
3. **Our method.** The top-1 video transition predicted by our method at each transition point is used as the best selection.

We collect 20 groups of video results in total for user study. Each group contains three videos edited using above three methods respectively. We invite overall 15 non-expert volunteers to participate in the evaluation. They are asked to choose a favorite video from each group (Q1) and rate each video on a scale of 1 to 5 (1 = poor, 5 = excellent, Q2), taking into account the general visual quality of videos and the matching degree between transitions and video/audio. Table 5 and Fig. 6 show the statistics of the results. As shown in Table 5, the videos

Table 5. The statistical results of user study. The inference time is reported as the time cost of our method.

Method	Avg. score	Avg. time (per video)
Weighted random pick	2.96	–
Professional video editor	3.80	7.5 min
Our method	3.76	1.5 s

Fig. 6. The voting results for three methods in Q1.

[1] https://www.videvo.net/.

generated using random pick receive the lowest score. Our method is pretty close to the professional editing in terms of average score, but being much more efficient by drastically reducing the average processing time of each video from 7.5 min to only 1.5 s (a **300×** speedup). Interestingly, in Fig. 6 which shows the voting results of Q1, one can see that videos from our methods are slightly more appealing than that from the professional editor. Above experimental results clearly demonstrate the advantages of our method in producing high-quality video transitions recommendations.

7 Conclusions and Future Works

The recent development of online video tools and platforms creates a high demand for a user-friendly video editing experience, which asks for a computational method or artificial intelligent model to lower the barrier, improve efficiency and ensure quality for doing video editing. Therefore, we propose a new task of video transition recommendation (VTR) to automatically recommend transitions based on any visual and audio inputs. We start with building a large-scale transition dataset. Then we formulate VTR as a multi-modal retrieval problem and propose a flexible framework for addressing the task. Through extensive qualitative and quantitative evaluations, we clearly demonstrate the effectiveness of our method. We hope this work can inspire more researchers to work on VTR and bring creativity and convenience to both professionals and non-professional users. Future works include but is not limited to extending the framework to support more video editing effects like video animation, 3D movements, etc., developing more efficient models for mobile deployment and integrating with other video editing techniques to create more comprehensive video editing systems.

Acknowledgment. Yaojie Shen did this work when interning at ByteDance Inc.

References

1. Frey, N., Chi, P., Yang, W., Essa, I.: Automatic non-linear video editing transfer. arXiv preprint arXiv:2105.06988 (2021)
2. Wang, M., Yang, G.W., Hu, S.M., Yau, S.T., Shamir, A.: Write-a-video: computational video montage from themed text. ACM Trans. Graph. **38**(6), 1–177 (2019)
3. Koorathota, S., Adelman, P., Cotton, K., Sajda, P.: Editing like humans: a contextual, multimodal framework for automated video editing. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Computer Society, pp. 1701–1709 (2021)
4. Liao, Z., Yu, Y., Gong, B., Cheng, L.: Audeosynth: music-driven video montage. ACM Trans. Graph. (TOG) **34**(4), 1–10 (2015)
5. Pardo, A., Caba, F., Alcázar, J.L., Thabet, A.K., Ghanem, B.: Learning to cut by watching movies. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6858–6868 (2021)
6. Hendricks, L.A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., Russell, B.: Localizing moments in video with natural language. In: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE Computer Society, pp. 5804–5813 (2017)

7. Akbari, H., et al.: Vatt: transformers for multimodal self-supervised learning from raw video, audio and text. In: Advances in Neural Information Processing Systems (2021)
8. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. IEEE Trans. Pattern Anal. Mach. Intell. **39**(4), 664–676 (2017)
9. Miech, A., Zhukov, D., Alayrac, J.B., Tapaswi, M., Laptev, I., Sivic, J.: Howto100m: learning a text-video embedding by watching hundred million narrated video clips. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 2630–2640. IEEE (2019)
10. Escorcia, V., Soldan, M., Sivic, J., Ghanem, B., Russell, B.: Temporal localization of moments in video collections with natural language. arXiv preprint arXiv:1907.12763 (2019)
11. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Advances in Neural Information Processing Systems, vol. 16 (2003)
12. Faghri, F., Fleet, D.J., Kiros, J.R., Fidler, S.: Vse++: improving visual-semantic embeddings with hard negatives. arXiv preprint arXiv:1707.05612 (2017)
13. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society pp. 815–823 (2015)
14. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Feragen, A., Pelillo, M., Loog, M. (eds.) SIMBAD 2015. LNCS, vol. 9370, pp. 84–92. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24261-3_7
15. Vaswani, A., et al.: Attention is all you need. In: Advances in neural information processing systems, vol. 30 (2017)
16. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: a video vision transformer. arXiv preprint arXiv:2103.15691 (2021)
17. Gabeur, V., Sun, C., Alahari, K., Schmid, C.: Multi-modal transformer for video retrieval. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12349, pp. 214–229. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58548-8_13
18. Lin, T., Wang, Y., Liu, X., Qiu, X.: A survey of transformers. arXiv preprint arXiv:2106.04554 (2021)
19. Hendricks, L.A., Mellor, J., Schneider, R., Alayrac, J.B., Nematzadeh, A.: Decoupling the role of data, attention, and losses in multimodal transformers. Trans. Assoc. Comput. Linguist. **9**, 570–585 (2021)
20. Lin, X., Bertasius, G., Wang, J., Chang, S.F., Parikh, D., Torresani, L.: Vx2text: end-to-end learning of video-based text generation from multimodal inputs. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, pp. 7001–7011 (2021)
21. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE Computer Society, pp. 6201–6210 (2019)
22. Won, M., Chun, S., Nieto, O., Serrc, X.: Data-driven harmonic filters for audio representation learning. In: ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 536–540. IEEE (2020)
23. Lei, J., Wang, L., Shen, Y., Yu, D., Berg, T.L., Bansal, M.: Mart: memory-augmented recurrent transformer for coherent video paragraph captioning. arXiv preprint arXiv:2005.05402 (2020)