The Anatomy of Video Editing: A Dataset and Benchmark Suite for AI-Assisted Video Editing

Dawit Mureja Argaw[1,2], Fabian Caba Heilbron[1], Joon-Young Lee[1],
Markus Woodson[1], and In So Kweon[2]

[1] Adobe Research
[2] KAIST

Abstract. Machine learning is transforming the video editing industry. Recent advances in computer vision have leveled-up video editing tasks such as intelligent reframing, rotoscoping, color grading, or applying digital makeups. However, most of the solutions have focused on video manipulation and VFX. This work introduces the Anatomy of Video Editing, a dataset, and benchmark, to foster research in AI-assisted video editing. Our benchmark suite focuses on video editing tasks, beyond visual effects, such as automatic footage organization and assisted video assembling. To enable research on these fronts, we annotate more than 1.5M tags, with relevant concepts to cinematography, from 196176 shots sampled from movie scenes. We establish competitive baseline methods and detailed analyses for each of the tasks. We hope our work sparks innovative research towards underexplored areas of AI-assisted video editing. Code is available at: https://github.com/dawitmureja/AVE.git.

1 Introduction

What does the future of video editing look like? Arguably AI-based technologies will have a strong influence on this creative industry. In fact, the computer vision community has already delivered technologies such as automatic rotoscoping [30] as a teaser of the opportunities to transform video editing. Most research development has centered around enabling AI-based VFX (visual effects) [30,23,16,10,26,33,38]; however, editing video involves more than that. Despite their importance towards AI-assisted video editing, topics such as understanding cinematography concepts for automatic organization and assisting editors to assembly edits remain underexplored in the computer vision community.

Research progress toward AI-assisted video editing has been hindered by the lack of formally defined tasks relevant to the editing process. This observation sparked recent works to study film properties or learn cutting patterns from movie data [14,44,31]. MovieNet [14] touches on cinematography style by providing annotations for two attributes: view scale and camera movement. While automatically tagging these concepts already provides value to the automatic organization, the cinematographic vocabulary comprises a much larger set. Learning to Cut [31] recommends the best moments to cut a pair of shots by looking at motion. This task is indeed an important editing task, but there are still other
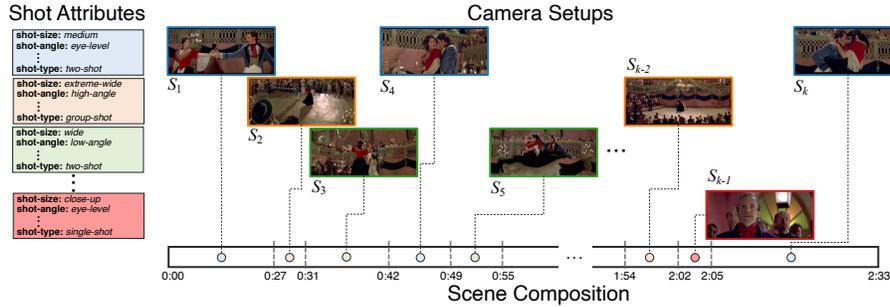
Fig. 1: Anatomy of Video Editing (AVE) dataset and benchmark suite. Our dataset decomposes movie scenes into sequence of shots by identifying the cuts in the scene. Each shot composing the scene has cinematographic attributes and camera setup labels.

decisions such as establishing the order of shots and the most suitable composition to assist video assembling. The progress is exciting, but the existing tasks cover a limited span of video editing.

To enable research development on AI-assisted video editing, we introduce the Anatomy of Video Editing (AVE), a dataset and benchmark suite. Movies require extensive hours of assistant editors to organize and tag footage and they depict the most creative and artistic forms of video editing. These properties motivate us to build the AVE dataset upon 5591 movie scenes. We recover the temporal composition of each scene by annotating the shot transitions and the camera setups. In total, we annotate 196176 shots among eight cinematography properties, yielding more than 1.5M labels. Fig. 1 illustrates one annotated movie scene from our dataset.

Our benchmark suite facilitates research in two areas to advance AI-assisted video editing. We define two tasks related to automatically organizing footage and introduce three tasks that aim to learn editors' patterns in video assembling. Equally crucial to defining the right tasks is establishing solid baselines, metrics, and initial analyses. Our baselines include modern video understanding methods, providing a competitive start. Nevertheless, our analyses discuss opportunities to develop new models to improve upon our baselines on the proposed tasks.

Contributions. To summarize, our contributions are two-fold:

(1) We introduce the AVE dataset, which includes the composition of 5591 movie scenes with more than 1.5M cinematography labels for 196176 shots (Sec. 3).

(2) We establish a benchmark suite that includes five different tasks for AI-assisted video editing (Sec. 4). Along with each task definition, we implement competitive baselines and provide extensive experimental analyses (Sec. 5).

2   Related Works

Movie Datasets. Several movie-based datasets have been presented by previous works [2,50,41,22,44,14] for various video understanding tasks. Zhu *et al.* [50] proposed the MovieBook dataset for aligning stories from books and

movies. Tapaswi *et al.* [41] introduced the `MovieQA` dataset. Bain *et al.* [2] proposed `Condensed Movies`, which contains 33K short movie clips and high-level text descriptions for text-to-video retrieval task. Liu *et al.* [22] proposed `MUSES`, which contains 31477 clips collected from several drama episodes and annotated with 25 action categories for temporal event localization task. Wu *et al.* [44] explored the long-term video understanding problem on movie clips dataset by formulating high-level prediction tasks. Recently, Huang *et al.* [14] proposed the `MovieNet`, which contains 1100 full movies and 60K trailers with diverse annotations to learn various tasks such as genre prediction, scene segmentation, and character recognition. While previous works primarily focus on high-level video content understanding tasks at a clip level, our work goes one step further and studies the sequence of shots in movie scenes by proposing a large-scale dataset with shot-level attributes and scene-level composition annotations.

Shot Assembly in Video Editing. Here we focus on previous works that studied cinematography patterns in film editing. For an in-depth discussion on video editing, in general, we invite the reader to [49]. Given a script and different takes of a scene, Leake *et al.* [20] attempted to generate a scene by selecting a relevant clip for each line of dialogue in a given script. Wu *et al.* [46,45] used shot relation attributes to formulate film editing pattern syntax and provided an interactive editing platform. Minh *et al.* [12] decomposed a video scene into an ordered sequence of relevant shots to remove abrupt discontinuities for improved human action recognition. Other works [3,35,31,32] studied cutting patterns in movie scenes. Although some previous works [45,46,20] attempted to study the film editing process, they are limited to a particular type of scene, *e.g.* dialogue, or cannot generalize to new editing patterns beyond a predefined set. Our work aims at learning general editing patterns from a large set of publicly-available movie scenes using a data-driven approach.

3   Anatomy of Video Editing: Dataset

Here we describe the collection of the Anatomy of Video Editing (`AVE`) dataset, a large-scale shot attribute set that contains approximately $196,176$ shots dissected from $5,591$ publicly available movie scenes [2,31][3].

3.1   Shot Attributes

Following the standard definition of shot properties in cinematography[28,7], we label eight attributes, which we define below, for each shot in `AVE`.

   Shot size is defined as how much of the setting or subject is displayed within a given shot. Shot size has *five* categories: 1) `Extreme wide` (`EW`) shots barely show the subject and the shot's main focus is the subject's surrounding; 2) `Wide` (`W`) shots, also known as long shot, show the entire subject and their relation to the surrounding environment; 3) `Medium` (`M`) shots depict the subject approximately from the waist up emphasizing both the subject and their surrounding; 4) `Close-up` (`CU`) shots are taken at a close range intended to show greater detail

---

[3] We crawled the movie scenes from the MovieClips YouTube Channel.

to the viewer; 5) `Extreme close-up` (`ECU`) shots frame a subject very closely where the outer portions of the subject are often cut off by the frame's edges.

Shot angle is the location where the camera is placed to take a shot. Shot angle has *five* categories: 1) `Aerial` (`A`) shot is captured from an elevated vantage point; 2) `Overhead` (`O`) shot is when the camera is placed directly above the subject; 3) `Eye level` (`EL`) shot is a shot where the camera is positioned directly at the subject's eye level; 4) `High angle` (`HA`) shot is when the camera points down on the subject from above; 5) `Low angle` (`LA`) shot is when the camera is positioned below the eye level and looks up at the subject.

Shot type refers to the composition of a shot in terms of the number of featured subjects and their physical relationship to each other and the camera. Shot type has *six* categories: 1) `Over-the-shoulder` (`OTS`) shot shows the main subject from behind the shoulder of another subject; 2) `Single` (`S`) shot captures one subject; 3) `Two` (`2`) shot has two subjects featured in the frame; 4) `Three` (`3`) shot has three characters in the frame 5) `Insert` (`I`) shot is any shot whose purpose is to draw the viewer's attention to a specific detail within a scene; 6) `Group` (`G`) shot features a group of subjects in the shot.

Shot motion is defined as the movement of the camera when taking a shot. Shot motion has *five* categories: 1) `Pan/Truck` (`P/T`) shot is when the camera is moving horizontally while its base remains in a fixed position; 2) `Tilt/Pedestal` (`T/P`) shot is when the camera moves vertically up or down with its base fixated to a certain point; 3) `Locked` (`L`) shot is taken without shifting the position of the camera; 4) `Zoom/Dolly` (`Z/D`) shot is when the camera moves forward and backward adding depth to a scene; 5) `Handheld` (`H`) shot is taken with the camera being supported only by the operator's hands and shoulder.

Shot location refers to the environment where the shot is taken. Shot location has *two* categories: 1) `Exterior` (`Ext`) shot is taken outdoors; 2) `Interior` (`Int`) shot is taken indoors.

Shot subject is the main subject featured or conveyed in the shot. Shot subject has *seven* categories: 1) `Animal`, 2) `Location`, 3) `Object`, 4) `Human`, 5) `Limb`, 6) `Face` and 7) `Text`.

Num. of people is the number of humans displayed in the shot and it has *six* categories: 1) `None` (`0`), 2) `One` (`1`), 3) `Two` (`2`), 4) `Three` (`3`), 5) `Four` (`4`) and 6) `Five` (`5`) if the shot has five or more people.

Sound source refers to the source of sound in the shot. Sound source has *four* categories: 1) `On screen` (`OnS`) - the source is a subject within the shot; 2) `Off screen` (`OfS`) - the sound comes from a subject not shown in the shot; 3) `External narration` (`EN`) - the source is a narration outside the shot; 4) `External music` (`EM`) - the only sound in the shot is music.

3.2   Scene Composition and Camera Setups

In addition to the attributes of the individual shots, we also provide annotation for the shot sequence composition, where we label the `start` and `end` time of each shot within a scene. We also group the shots that belong to the same `camera setup` and annotate the total number of takes used in the edited scene. These annotations will enable our studies on shot pattern selection and sequencing.

Table 1: Comparison with related datasets.

	Num. of Shots	Num. of Videos	Num. of Shot Attributes	Scene Composition	Camera Setup
Lie 2014 [4]	327	327	1	✗	✗
Context 2011 [48]	3,206	4	1	✗	✗
Cinema 2013 [5]	3,000	12	1	✗	✗
Taxon 2009 [43]	5054	7	1	✗	✗
MovieSD 2020 [35]	46,857	7,858	2	✗	✗
AVE	196,176	5,591	8	✓	✓

Table 2: Statistics of AVE.

	Train	Val	Test	Total
Num. of scenes	3914	559	1,118	5,591
Num. of shots	151,053	15,040	30,083	196,176
Avg. duration of shots (sec.)	3.83	3.71	3.78	3.81
Avg. number of shots per scene	34.71	35.42	35.69	35.09
Avg. number of camera setups	5.71	6.11	5.69	5.74

3.3   Annotation Procedure

We recruited a task force of 15 professional video editors. To reduce the amount of manual effort, we pre-segmented each video (scene) into shots using a pre-trained shot-boundary detector [39]. Then, the annotation process consisted of two steps. First, we asked the annotators to verify the automatic shot boundaries are correct and to group the shots in a scene by camera setup. Second, we asked the task force to label each shot with the attributes listed in Sec. 3.1.

3.4   Dataset Statistics

The Anatomy of Video Editing (AVE) dataset consists of 196,176 holistically annotated shots, collected from 5,591 movie scenes that cover a wide range of genres. In Table 1, we compare AVE with related datasets. Our dataset is considerably larger in size with significantly more comprehensive, and relevant to video editing, annotations. Previous works primarily focus on individual shot properties to analyze certain cinematic techniques, but AVE goes beyond shot level attributes by offering the temporal sequencing of shots and the composition of scenes. Table 2 presents detailed statistics for our train-val-test splits.

4   Anatomy of Video Editing: Benchmark Suite

In this section, we introduce five tasks for AI-assisted Video editing. The first two tasks focus on benchmarking the ability to automatically organize and tag footage according to cinematography properties. The last three tasks center around predicting editing patterns used in movie scenes.

Notation. Let $\mathcal{M}$ represent a movie scene which is defined as a sequence of k shots, i.e. $\mathcal{M} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k\}$, where $\mathcal{S}$ denotes a shot clip. Each shot $\mathcal{S}$ is composed of visual (v) and audio (a) representations, i.e. $\mathcal{S} = \langle v, a \rangle$. The audio-visual features encoded from each shot clips are denoted as $\{u_1, u_2, \ldots, u_k\}$.
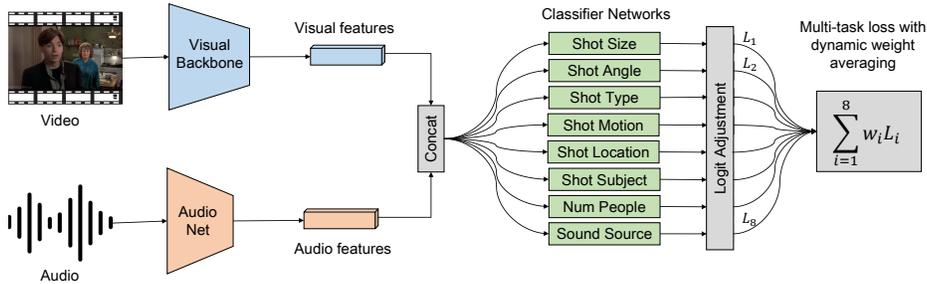
Fig. 2: Overview of shot attributes classification framework. Given a shot clip, we first extract audio-visual features using a common backbone network. The extracted features are then feed into eight classifiers to predict the attributes of the shot.

4.1   Shot Attributes Classification

Given a shot $\mathcal{S}$, shot attributes classification aims to predict the attributes of $\mathcal{S}$: `shot size`, `shot angle`, `shot type`, `shot motion`, `shot location`, `shot subject`, `number of people` and `sound source` as discussed in Sec. 3.1. This task would be useful for an editor to automatically classify and organize shots by their attributes during the editing process. It can also be coupled with other tasks such as video content understanding [14,2,44,22] to establish a user query system, where shots can be retrieved by their content and attributes.

As each attribute has its own classes independent of others, shot attributes classification can be defined in a *multi-task* setting, where multiple classifiers are jointly optimized together in an end-to-end manner. We design a general framework for shot attributes classification by cascading an audio-visual encoder network with eight classifier networks as shown in Fig. 2. We use `ResNet-101` [11] and `R-3D` [42] as visual backbone networks to extract features from image and video inputs, respectively. To incorporate features from the audio input, we design a network called `AudioNet`, which is a feed-forward network with 3 convolutional and 2 linear layers. The features extracted from different input representations are then cascaded to obtain a cross-modal feature via concatenation. This cross-modal feature is then fed into each classifier network which outputs the predicted class for the respective shot attribute. Each classifier is a simple network with 2 linear layers and ReLU activation units.

Shot attributes inherently exhibit a long tail label distribution. For instance, *medium* is the most common `shot size` in movie scenes while *extreme close-up* is hardly used. This imbalance makes naive training to be biased toward the dominant label [27,15,40]. To address this problem, we implement the idea of logit adjustment [27] on each classifier output according to the label frequencies in the respective shot attribute. The training loss for our network is defined as an aggregate of the cross entropy losses from the different classifiers. To better explore the cross-task correlation during training, we use dynamic weight averaging technique [21] to scale the loss of each task (attribute) during training.
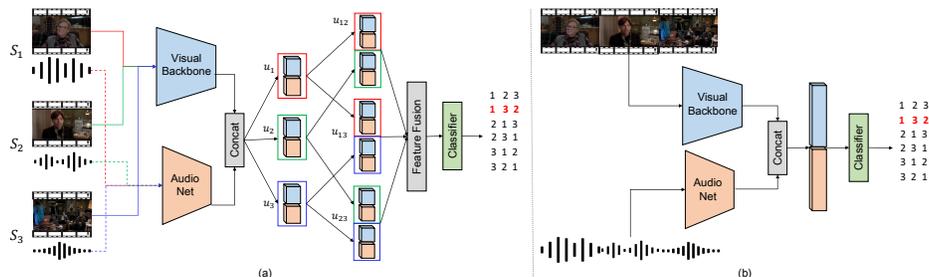
Fig. 3: Shot ordering baselines. (a) Late feature fusion, where we first extract features from each shot in the sequence and then combine the features at later stage (b) Early input fusion, where the input shot clips are first concatenated before extracting feature.

4.2 Camera Setup Clustering

Movie scenes in general contain highly frequent shot cuts as they are professionally edited by connecting several shots captured using a multi-camera system. This phenomenon can also be observed in the proposed dataset (see Table 2) which contains approximately 35 shots and 6 camera setups per scene. Given a list of shots $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k\}$ in any order, shot clustering is defined as grouping shots that belong to the same camera setup. This task could be useful during the editing process in order to catalog different shots of a scene or various takes of a particular shot into the respective camera setup they belong to.

We formulate this task as a high-dimensional feature clustering problem. We extract features from the given shot set and evaluate the performance of several state-of-the-art clustering algorithms. We use both traditional, *i.e.* SIFT [25], and learning-based, *i.e.* ResNet-101 [11], CLIP [34] and R-3D [42], feature extraction methods. To establish baselines for shot clustering task, we experiment with standard clustering algorithms such as K-Means [24], Hierarchical Agglomerative Clustering (HAC) [29], OPTICS [1], but also novel methods such as FINCH [36].

4.3 Shot Sequence Ordering

Shots are fundamental units in the filmmaking process. Film editors create scenes by assembling shots in a coherent pattern that best depicts the story of the scene. As previous studies [3,46,45,20] have indicated, several factors go into the selection and sequencing of shots, which can be highly subjective at times. In this task, we aim to learn general shot ordering patterns in movie scenes following a data-driven approach, where we break down a movie scene into shots, randomly shuffle the shots, and target to reconstruct the movie scene by reordering the shuffled shots. Given a sequence of *contiguous* but randomly shuffled shots, *i.e.* rand$\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k\}$, shot ordering can be formulated as a classification task. If a given scene has k shots, there are $k!$ (factorial of k) possible ways of ordering them, *i.e.* $k!$ classes. We set $k = 3$ at a time for convenience and define shot order prediction as a 6-way classification problem.

We experiment with two types of baselines for shot order prediction as shown in Fig. 3. First, we follow previous video representation learning works [48,47]
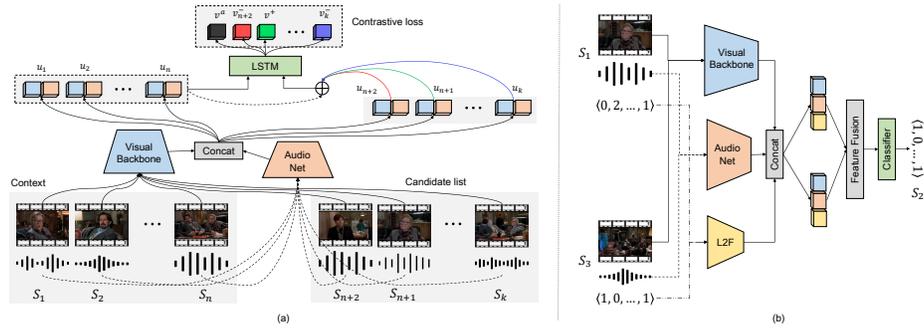
Fig. 4: (a) Overview of next shot selection. We use a contrastive approach to learn positive (matching) and negative (non-matching) shot sequence patterns. (b) Missing shot attributes prediction. We predict the attributes of an intermediate shot given its left and right neighboring shots.

and perform late feature fusion, where we extract features from the input shot clips and then perform a hierarchical fusion of features. The combined features are then passed to a classifier network to predict the order (see Fig. 3a). As a second baseline, we perform early fusion at an input level where we concatenated the shot clips and then extract features from the resulting input (see Fig. 3b).

Compared to existing works on video clips order prediction [48,47,9,19], shot ordering is a more challenging problem for two main reasons. First, previous works mostly deal with ordering different segments of a video with a single camera setup. This makes it convenient to exploit semantic and geometric correspondences across clips to analyze their temporal coherence. In contrast, the neighboring clips in a given shot sequence are often from different camera setups and there is much less content overlap across inputs making it very challenging to learn ordering patterns from only the visual stream. Second, in previous works [48,47,9,19] problem formulation, there is always a unique solution for ordering the input clips given that the interval between the clips is not too large. In comparison, due to the subjective (and artistic) nature of the task, there could be multiple ways of ordering shot clips in a movie scene [3,46,45,20].

4.4   Next Shot Selection

During film editing, the process of assembling shots occurs in a sequential manner. Given a partial sequence of shots as a context, this task aims to anticipate the next shot from the list of available shots. Let $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\}$ denote the sequence of n shots provided as a history and $\texttt{rand}\{\mathcal{S}_{n+1}, \mathcal{S}_{n+2} \ldots, \mathcal{S}_k\}$ represent the list of $k - n$ possible candidates to follow $\mathcal{S}_n$. Next shot selection is then defined as a multiple choice problem where a model makes the decision based on the affinity of each candidate shot to the previous sequence.

We formulate this task following a contrastive learning approach. First, we extract features from each shot in the given context and candidate list using an `audio-visual encoder` network (see Fig. 4). It is important to learn the

shot sequence pattern from the given context in order to anticipate the next shot candidate. Thus, we feed the extracted feature sequence $\{u_1, u_2, ..., u_n\}$ into an LSTM [13] module, *i.e.* $v^a = \text{LSTM}\{u_1, u_2, \ldots, u_n\}$. The output of the LSTM module is defined as an *anchor* feature v^a which represents an embedding for the context sequence. To generate a positive (matching) feature v^+ during training, we cascade the context and the correct next shot $\mathcal{S}_{n+1}$ and input the resulting sequence into the LSTM module, *i.e.* $v^+ = \text{LSTM}\{u_1, u_2, \ldots, u_n, u_{n+1}\}$. The intuition here is to learn that $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\}$ and $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n, \mathcal{S}_{n+1}\}$ are feasible shot sequence patterns that appear in actual movie scenes. In contrast, the negative (non-matching) features are created by cascading any shot **except** $\mathcal{S}_{n+1}$ to the context, *i.e.* $\{v_i^-\}_{i=1} = \text{LSTM}\{u_1, u_2, \ldots, u_n, u_i\}$, where $i \neq n + 1$. We experiment with two types of negative samples generation: i) `in-sequence`, where $k - n - 1$ negatives are sampled per each input using all incorrect choices in the respective candidate list and ii) `in-batch`, where all other shot sequences in the batch are additionally used, *i.e.* $bk - n - 1$ negative samples per each input for a batch size of b. We use the supervised NT-Xent loss [6,18] to train our network. During inference, we compute the affinity score, the cosine similarity, between the anchor feature v^a and the feature $v^c = \text{LSTM}\{u_1, u_2, \ldots, u_n, u_c\}$, where c is an index for a shot in the candidate list. The shot with the highest score is then selected as the next shot.

4.5   Missing Shot Attributes Prediction

With the goal of learning editing patterns in movie scenes, here, we define another task that aims to predict the attributes of a missing shot in a given incomplete shot sequence. This is different from the task in Sec. 4.4 as it targets to predict the likely attributes of the shot that best completes the given sequence irrespective of the shot availability. Let $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k\}$ denote a given incomplete sequence without $\mathcal{S}_i$, where $1 < i < k$. Then, missing shot attribute prediction is formulated as a classification problem, where we predict the attributes of $\mathcal{S}_i$ using the input sequence of shots and their attributes as a context. In this work, we consider a simple setup with $k = 3$, *i.e.* $\{\mathcal{S}_1, \mathcal{S}_3\}$ is given as an input and we predict the attributes of $\mathcal{S}_2$. A more generalized formulation with longer sequences and missing shot(s) at arbitrary time steps is left for future work.

Fig. 4b depicts the framework for missing shot attribute prediction. We first extract features from $\mathcal{S}_1$ and $\mathcal{S}_3$ using a pretrained backbone network. The attributes of $\mathcal{S}_1$ and $\mathcal{S}_2$ are also added as an input. For this purpose, we designed a simple 1-layered linear network called `label-2-feature` (L2F) which transforms an attribute vector of size 8 into a feature embedding. The extracted features are then concatenated and fed into multiple classifier networks which predict the different attributes of the missing shot $\mathcal{S}_2$. Like the task in Sec. 4.1, logit adjustment is applied to the output of the classifiers during training to prevent the network from being biased to the dominant labels. The training is done in a multi-task setting using dynamic weight averaging [21] to scale the cross entropy losses from the different classifiers.

Table 3: Quantitative analysis on shot attributes classification.

Attribute	Multi-task training								Single-task training			
	Video				Video + Audio				Video + Audio			
	Naïve		Logit adj.		Naïve		Logit adj.		Naïve		Logit adj.	
	Val	Test	Val	Test	Val	Test	Val	Test	Val	Test	Val	Test
Shot size	35.7	35.5	67.8	66.9	36.2	36.4	66.8	65.0	38.7	39.1	70.9	67.6
Shot angle	25.8	25.8	62.2	53.2	27.6	27.7	58.6	49.5	29.1	28.9	63.0	49.8
Shot type	59.5	60.8	63.9	64.9	59.8	61.0	63.7	65.3	60.1	62.3	64.9	66.7
Shot motion	32.1	31.7	42.8	42.7	32.3	31.8	44.6	43.2	32.3	31.2	47.4	43.7
Shot location	82.9	81.9	84.4	83.3	83.0	80.9	83.7	83.7	83.4	82.8	83.9	84.0
Shot subject	40.0	39.8	50.8	47.4	40.0	39.7	50.2	46.7	42.2	40.7	54.2	48.0
Num. of people	55.0	55.1	61.3	61.2	55.1	55.3	60.9	61.4	56.1	57.1	61.5	63.5
Sound source	25.0	25.0	34.4	32.6	25.0	25.0	41.0	38.9	25.0	25.0	38.1	35.3
Average	44.5	44.4	**58.4**	**56.5**	44.9	44.7	**58.7**	**56.7**	45.9	45.9	**60.5**	**57.3**

5 Experimental Results and Discussion

Dataset. We follow a `train-val-test` scene split of 70-10-20 in all experiments (see Table 2). As the scenes in the proposed dataset are non-overlapping, the train, validation, and test splits are disjoint sets. For the shot attributes classification task, we use all the shots in the respective scene split for training and evaluation. For shot ordering and missing shot attributes prediction tasks, we generate train, validation, and test sets by sampling 3 consecutive shots from a scene at a time. For the next shot selection task, on the other hand, we sample 9 consecutive shots at a time. The first 4 shots in the sequence are used as a context. The remaining 5 shots are used to make a candidate list.

Evaluation metrics. For shot attributes classification and missing shot attributes prediction tasks, we report the `average per-class accuracy` to take the long tail distribution problem into account. For shot ordering and next shot selection tasks, we simply evaluate the `overall accuracy`. For shot clustering task, we evaluate the quality of the generated clusters with respect to ground truth clusters on 3 different metrics: Purity score (`PS`), Normalized mutual information (`NMI`) and Rand index (`RI`) [37].

Implementation Details. We use `ResNet-101` [11] and `R-3D` [42] as visual backbone networks to extract features from image and video inputs, respectively. In all experiments, the backbone network is initialized with pretrained weights (`ResNet-101` - pretrained on ImageNet [8] and `R-3D` - pretrained on Kinetics-400 [17]) and fine-tuned during training. We uniformly sample 16 frames from a shot clip as an input to `R-3D`, while we use the central frame of a shot for `ResNet-101`. Refer to the supplementary for task-level implementation details.

5.1 Experimental Results

Shot Attributes Classification. In Table 3, we present the performance of our network trained in two different settings: i. `multi-task`, where all eight classifiers are jointly trained together in an end-to-end manner, and ii. `single-task`, where one classifier is optimized at a time. As can be inferred from Table 3, individually training each classifier generally results in a better accuracy compared to training all classifiers together. It can also be noticed from Table 3 that taking the long

Table 4: Quantitative analysis on camera setup clustering.

| | Parameter-based | | | | | | Parameter-free | | | | | |
| | K-Means [24] | | | HAC [29] | | | OPTICS [1] | | | FINCH [36] | | |
Method	RI	NMI	PS	RI	NMI	PS	RI	NMI	PS	RI	NMI	PS
SIFT [25]	0.863	0.779	0.837	0.865	0.785	0.840	0.786	0.685	0.790	0.795	0.692	0.777
ResNet-101 [11]	**0.947**	**0.913**	**0.937**	**0.946**	**0.914**	**0.939**	0.784	0.718	0.801	0.887	0.834	0.887
CLIP [34]	0.907	0.858	0.894	0.912	0.867	0.899	0.836	0.769	0.865	0.845	0.780	0.851
ResNet-101 (Ours)	0.921	0.873	0.908	0.922	0.876	0.909	**0.868**	**0.802**	**0.889**	**0.889**	**0.838**	**0.895**
R-3D [42]	0.906	0.846	0.891	0.910	0.856	0.897	0.693	0.601	0.723	0.814	0.728	0.813
R-3D (Ours)	0.902	0.840	0.882	0.903	0.844	0.884	0.850	0.775	0.871	0.872	0.805	0.866

tail distribution problem into account during training consistently leads to a significantly better result in comparison with naïve training. For example, in a single-task setting, applying logit adjustment (Logit adj.) during training results in a 31.8% and 24.8% average accuracy improvement on validation and test sets, respectively. For attributes with imbalanced label distributions such as `shot size` and `shot angle`, we have observed that naïvely trained network performs very well for the dominant classes but extremely poorly for low-frequency classes. On the other hand, a network trained with logit adjustment gives a relatively balanced per-class accuracy, and hence better overall performance.

We use different types of input representations for shot attributes classification. Table 3 compares their performance. Using video features as an input gives a competitive performance for most shot attributes except for `sound source`, where adding audio features results in a 19.3% performance boost on both validation and test sets. Our baselines achieve a relatively low accuracy when predicting `shot motion` and `sound source`. The lack of explicit modeling of motion could be one contributing factor to the low performance on `shot motion` [35]. The low accuracy `sound source` classification is most likely due to the fine-grained nature of the task. For instance, it could be very ambiguous to differentiate between `on-screen`, `off-screen` and `external-narration` classes. Incorporating motion information in the form of optical flow along with other input modalities and exploring the correlation of audio and visual features with a carefully designed attention mechanism are interesting research directions.

Camera Setup Clustering. We perform scene-level shot clustering, where we group the shots of a given scene into different camera setups. The averaged results for all scenes in the dataset are summarized in Table 4. We use image-based and video-based feature extraction methods to establish baselines. We also compare the visual backbone of our framework trained on shot attribute prediction task in Sec. 3.1. For clustering, we experiment with four standard clustering algorithms. `K-Means` [24] and `HAC` [29] require the number of clusters as an input parameter, *i.e. parameter-based*, while `OPTICS` [1] and `FINCH` [36] generate clusters without relying on the number of clusters as an input, *i.e. parameter-free.*

Table 4 shows that image-based feature extraction methods generally perform better than video-based backbones. It is also worth noting that, for parameter-based clustering, `ResNet-101` [11] pretrained on ImageNet [8] achieves the highest clustering accuracy on all metrics. However, for parameter-free cluster-

Table 5: Quantitative analysis on shot sequence ordering.

Method	Frame			Video			Audio + Video		
	Val	Test	Survey	Val	Test	Survey	Val	Test	Survey
Random	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6
Baseline-I	21.0	21.5	20.0	21.7	22.4	23.3	23.1	24.4	26.7
Baseline-II	25.0	25.7	26.7	26.5	27.4	33.3	**29.3**	**30.7**	33.3
Human	-	-	32.0	-	-	39.9	-	-	**55.6**
Cinematography patterns									
Baseline-I (insert)	26.2	25.8	-	27.9	27.5	-	30.1	29.8	-
Baseline-II (insert)	39.6	37.3	-	41.5	38.4	-	42.2	39.5	-
Baseline-I (intensify)	29.6	30.1	-	30.7	31.4	-	32.2	33.1	-
Baseline-II (intensify)	34.1	35.8	-	38.1	40.5	-	44.2	48.0	-

ing, `ResNet-101` pretrained on the proposed dataset consistently outperforms `ResNet-101` (pretrained on ImageNet). The same notion can also be observed for `R-3D` [42] which is pretrained on Kinetics-400 [17]. These results highlight that shot attributes classification could be used as a pretext task for better shot clustering as the ground truth for the number of clusters is not generally available.

Shot Sequence Ordering. The results for the shot order prediction task are presented in Table 5. We evaluate two baselines for shot order prediction: i. `Baseline-I`, where we first extract features from each shot in the sequence and then perform hierarchical feature fusion in the later stage as shown in Fig. 3a, and ii. `Baseline-II`, where we first concatenate the shots in the given input and then extract features from the resulting sequence (see Fig. 3b). We evaluate the two baselines for different input representations. As shot ordering is a 6-way classification problem, random order prediction has an accuracy of 16.6%. As can be seen from Table 5, the best performance for both baselines is achieved when using `audio + video` compared to using only `video` or a single `frame` as an input. This is intuitive as audio-visual features provide a richer context for the network to find correspondence between the shots when predicting order.

Table 5 also shows that early fusion of inputs leads to significantly better results compared to late feature fusion. For example, when using `audio + video`, `Baseline-II` outperforms `Baseline-I` by a margin of 26.8% and 25.8% on validation and test sets, respectively. This is mainly because early fusion enables the model to implicitly learn the correlation between shots at different levels of abstraction since all inputs are processed simultaneously. Instead, late feature fusion only learns limited correspondences as each shot is encoded independently.

The results in Table 5 are low mainly because predicting the order of shots is a very challenging problem. As discussed in Sec. 4.3, the shots in the input sequence are often from different camera setups, and hence, it is very difficult to exploit semantic and geometric correspondences which are crucial to learning order. To further analyze the performance of our models in comparison with humans, we conducted a survey. We sampled 30 triplets from the test set and asked more than 160 people to predict the order of the randomly shuffled shots, where each shot is represented using a single `frame`, `video` and `video + audio`.

Table 6: Quantitative analysis on next shot selection.

Method	Frame		Video		Audio + Video	
	Val	Test	Val	Test	Val	Test
Random	20.0	20.0	20.0	20.0	20.0	20.0
Cosine Sim.	14.3	13.4	13.3	13.4	-	-
Ours (in-sequence)	34.1	34.0	37.9	37.5	39.0	38.7
Ours (in-batch)	38.4	38.2	41.3	41.0	**41.6**	**41.4**

To prevent people from exploiting very noticeable transitions between shots, we embedded blank frames between the shot clips.

As can be concluded from Table 5, despite the better accuracy compared to our baselines, the task of ordering shots is also difficult for humans particularly when the shots are represented using a `frame` or `video`. The significant performance surge for humans in `audio + video` setting is most likely because humans can comprehend the content of the audio. For instance, if there is a dialogue between subjects in the given shots, humans can easily establish the order of the shots based on the speech of the subjects. An exciting future work would be to further exploit the content of the audio in the form of speech or other relevant representations in order to imitate human comprehension.

Although a broadly formulated shot ordering problem could be challenging, we analyze the performance of our baselines for shot sequences that contain commonly used cinematography patterns [3,46,45,20]. First, we evaluate the `insert` pattern, where one of the shots in the sequence is an `insert` shot. In this case, the number of possibilities for ordering decreases to 2 since the `insert` shot should always be in the middle. It is worth noting that a model doesn't have this pre-existing knowledge. As can be seen from Table 5, the performance of both baselines significantly increases for validation and test samples that contain an `insert` shot. The same phenomenon can also be observed for `intensify` pattern, where an editor uses a sequence of shots moving gradually closer, *i.e.* decreasing `shot size`, to build up emotion [4]. These results highlight that our baselines have implicitly learned common cinematography patterns during training.

Next Shot Selection. Table 6 shows the results for next shot selection task. We experiment with a shot sequence length of 9 for quantitative evaluation, where the first 4 shots are used as context and the remaining 5 are randomly shuffled and fed into the network as a candidate list for the next shot. In this setup, the random chance of accurately selecting the next shot is 20%. To further demonstrate the previously mentioned concept that the neighboring shots in a movie scene are usually from different camera setups, we use a naïve `cosine similarity` between the end shot in the given context, *i.e.* $\mathcal{S}_4$, and each shot in the candidate list as a baseline for next shot selection task. Here, we extract features from the shots using pretrained backbone networks and evaluate the cosine similarity between the extracted features. As can be seen from Table 6, naïve `cosine similarity` performs even worse than random chance. In

---

[4] We consider 3 `intensify` patterns: `extreme-wide` - `wide` - `medium`, `wide` - `medium` - `close-up`, `medium` - `close-up` - `extreme-close-up`.

Table 7: Quantitative analysis on missing shot attributes prediction.

Method	Shot size		Shot angle		Shot type		Shot motion	
	Val	Test	Val	Test	Val	Test	Val	Test
Dominant label	20.0	20.0	20.0	20.0	16.7	16.7	20.0	20.0
Frame	40.9	32.4	22.6	30.5	26.6	26.1	25.0	25.8
Frame + Attributes	**47.8**	**44.6**	28.5	34.1	32.0	34.5	31.0	31.8
Video	34.3	32.8	29.0	34.7	31.3	33.2	30.1	30.7
Video + Attributes	38.3	35.3	30.4	37.2	32.7	35.0	31.9	32.1
Video + Audio	36.4	35.3	31.5	35.8	31.9	33.4	30.5	30.9
Video + Audio + Attributes	39.2	37.2	**33.7**	**39.0**	**32.9**	**35.4**	**32.3**	**33.2**

comparison, our proposed baselines (Sec. 4.4) perform significantly better. Our approach achieves an accuracy of 41.6% and 41.4% on validation and test sets, respectively. We also observe that using a larger number of negatives (`in-batch`) during training improves performance.

Missing Shot Attributes Prediction. This task targets to predict the attributes of an intermediate shot given its left and right neighboring shots. In Table 7, we present the results on four shot attributes, *i.e.* `shot size`, `shot angle`, `shot type` and `shot motion`, for a model trained in a multi-task setting. To confirm that the proposed model indeed uses the input shots as a context and does not simply converge to always predicting the dominant labels, we evaluated the accuracy of predicting the dominant label every time for each shot attribute. As can be inferred from the table, the proposed model outperforms the naïve `dominant label` prediction baseline by a large margin. It can also be noticed that incorporating the attributes of the input shots along with other representations consistently improves model accuracy across all attributes.

As can be inferred from Table 7, the multi-task training setup does not always lead to a balanced performance for the missing shot attributes prediction task. For instance, when using `frame` as an input, the performance gap between `shot size` and other attributes is notably large in comparison with using other input representations. This is mainly because the model overfitted to the `shot size` attribute for this particular input setup.

6   Conclusion

We introduced the Anatomy of Video Editing (AVE) dataset and benchmark. We gathered more than 1.5M manually labeled tags, with relevant concepts to cinematography, from 196176 shots sampled from movie scenes. We also annotated the shot transitions and camera setup in movie scenes, which allowed us to recover the scene composition. We also define five tasks to help attain research progress in automatic footage organization and assisted video assembling. We hope that our work will inspire new computer vision technologies and spur research in machine listening, speech and language understanding, and graphics. Moreover, we believe our dataset will foster the design of new relevant tasks for AI-assisted editing. For instance, our sound-source annotations can facilitate the study of music selection for video. The scene composition labels can enable tasks related to recommending pace and rhythm for cutting.

References

1. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: Ordering points to identify the clustering structure. ACM Sigmod record **28**(2), 49–60 (1999) 7, 11
2. Bain, M., Nagrani, A., Brown, A., Zisserman, A.: Condensed movies: Story based retrieval with contextual embeddings (2020) 2, 3, 6
3. Baxter, M.: Comparing cutting patterns–a working paper. Present webpage, Question **3** (2013) 3, 7, 8, 13
4. Bhattacharya, S., Mehran, R., Sukthankar, R., Shah, M.: Classification of cinematographic shots using lie algebra and its application to complex event recognition. IEEE Transactions on Multimedia **16**(3), 686–696 (2014) 5
5. Canini, L., Benini, S., Leonardi, R.: Classifying cinematographic shot types. Multimedia tools and applications **62**(1), 51–73 (2013) 5
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020) 9
7. Dancyger, K.: The technique of film and video editing: history, theory, and practice. Routledge (2018) 3
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009) 10, 11
9. El-Nouby, A., Zhai, S., Taylor, G.W., Susskind, J.M.: Skip-clip: Self-supervised spatiotemporal representation learning by future clip order ranking. arXiv preprint arXiv:1910.12770 (2019) 8
10. Gao, C., Saraf, A., Huang, J.B., Kopf, J.: Flow-edge guided video completion. In: European Conference on Computer Vision. pp. 713–729. Springer (2020) 1
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 6, 7, 10, 11
12. Hoai, M., Zisserman, A.: Thread-safe: Towards recognizing human actions across shot boundaries. In: Asian Conference on Computer Vision. pp. 222–237. Springer (2014) 3
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997) 9
14. Huang, Q., Xiong, Y., Rao, A., Wang, J., Lin, D.: Movienet: A holistic dataset for movie understanding. In: The European Conference on Computer Vision (ECCV) (2020) 1, 2, 3, 6
15. Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., Kalantidis, Y.: Decoupling representation and classifier for long-tailed recognition. arXiv preprint arXiv:1910.09217 (2019) 6
16. Kasten, Y., Ofri, D., Wang, O., Dekel, T.: Layered neural atlases for consistent video editing. ACM Transactions on Graphics (TOG) **40**(6), 1–12 (2021) 1
17. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017) 10, 12
18. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 18661–18673. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf 9

19. Kim, D., Cho, D., Kweon, I.S.: Self-supervised video representation learning with space-time cubic puzzles. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 8545–8552 (2019) 8
20. Leake, M., Davis, A., Truong, A., Agrawala, M.: Computational video editing for dialogue-driven scenes. ACM Trans. Graph. **36**(4), 130–1 (2017) 3, 7, 8, 13
21. Liu, S., Johns, E., Davison, A.J.: End-to-end multi-task learning with attention. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1871–1880 (2019) 6, 9
22. Liu, X., Hu, Y., Bai, S., Ding, F., Bai, X., Torr, P.H.: Multi-shot temporal event localization: a benchmark. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12596–12606 (2021) 2, 3, 6
23. Liu, Y.L., Lai, W.S., Yang, M.H., Chuang, Y.Y., Huang, J.B.: Learning to see through obstructions with layered decomposition. IEEE Transactions on Pattern Analysis & Machine Intelligence (01), 1–1 (2021) 1
24. Lloyd, S.: Least squares quantization in pcm. IEEE transactions on information theory **28**(2), 129–137 (1982) 7, 11
25. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004) 7, 11
26. Lu, E., Cole, F., Dekel, T., Zisserman, A., Freeman, W.T., Rubinstein, M.: Omnimatte: associating objects and their effects in video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4507–4515 (2021) 1
27. Menon, A.K., Jayasumana, S., Rawat, A.S., Jain, H., Veit, A., Kumar, S.: Long-tail learning via logit adjustment. arXiv preprint arXiv:2007.07314 (2020) 6
28. Metz, C.: Film language: A semiotics of the cinema. University of Chicago Press (1991) 3
29. Müllner, D.: Modern hierarchical, agglomerative clustering algorithms. arXiv preprint arXiv:1109.2378 (2011) 7, 11
30. Oh, S.W., Lee, J.Y., Xu, N., Kim, S.J.: Video object segmentation using space-time memory networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9226–9235 (2019) 1
31. 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) 1, 3
32. Pardo, A., Heilbron, F.C., Alcázar, J.L., Thabet, A., Ghanem, B.: Moviecuts: A new dataset and benchmark for cut type recognition. arXiv preprint arXiv:2109.05569 (2021) 3
33. Patwardhan, K.A., Sapiro, G., Bertalmío, M.: Video inpainting under constrained camera motion. IEEE Transactions on Image Processing **16**(2), 545–553 (2007) 1
34. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning. pp. 8748–8763. PMLR (2021) 7, 11
35. Rao, A., Wang, J., Xu, L., Jiang, X., Huang, Q., Zhou, B., Lin, D.: A unified framework for shot type classification based on subject centric lens. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16. pp. 17–34. Springer (2020) 3, 5, 11
36. Sarfraz, S., Sharma, V., Stiefelhagen, R.: Efficient parameter-free clustering using first neighbor relations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8934–8943 (2019) 7, 11

37. Schütze, H., Manning, C.D., Raghavan, P.: Introduction to information retrieval, vol. 39. Cambridge University Press Cambridge (2008) 10
38. Smith, J.R., Joshi, D., Huet, B., Hsu, W., Cota, J.: Harnessing ai for augmenting creativity: Application to movie trailer creation. In: Proceedings of the 25th ACM international conference on Multimedia. pp. 1799–1808 (2017) 1
39. Souček, T., Lokoč, J.: Transnet v2: An effective deep network architecture for fast shot transition detection. arXiv preprint arXiv:2008.04838 (2020) 5
40. Tan, J., Wang, C., Li, B., Li, Q., Ouyang, W., Yin, C., Yan, J.: Equalization loss for long-tailed object recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11662–11671 (2020) 6
41. Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., Fidler, S.: Movieqa: Understanding stories in movies through question-answering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4631–4640 (2016) 2, 3
42. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 6450–6459 (2018) 6, 7, 10, 11, 12
43. Wang, H.L., Cheong, L.F.: Taxonomy of directing semantics for film shot classification. IEEE transactions on circuits and systems for video technology **19**(10), 1529–1542 (2009) 5
44. Wu, C.Y., Krahenbuhl, P.: Towards long-form video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1884–1894 (2021) 1, 2, 3, 6
45. Wu, H.Y., Christie, M.: Analysing cinematography with embedded constrained patterns. In: WICED-Eurographics Workshop on Intelligent Cinematography and Editing (2016) 3, 7, 8, 13
46. Wu, H.Y., Palù, F., Ranon, R., Christie, M.: Thinking like a director: Film editing patterns for virtual cinematographic storytelling. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) **14**(4), 1–22 (2018) 3, 7, 8, 13
47. Xiao, J., Li, L., Xu, D., Long, C., Shao, J., Zhang, S., Pu, S., Zhuang, Y.: Explore video clip order with self-supervised and curriculum learning for video applications. IEEE Transactions on Multimedia **23**, 3454–3466 (2021). https://doi.org/10.1109/TMM.2020.3025661 7, 8
48. Xu, M., Wang, J., Hasan, M.A., He, X., Xu, C., Lu, H., Jin, J.S.: Using context saliency for movie shot classification. In: 2011 18th IEEE International Conference on Image Processing. pp. 3653–3656. IEEE (2011) 5, 7, 8
49. Zhang, X., Li, Y., Han, Y., Wen, J.: Ai video editing: a survey (2022) 3
50. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S.: Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In: Proceedings of the IEEE international conference on computer vision. pp. 19–27 (2015) 2