Learning Where to Cut from Edited Videos

Yuzhong Huang[*]
University of Southern California
Information Sciences Institute
yuzhongh@isi.edu

Xue Bai Oliver Wang Fabian Caba Aseem Agarwala
Adobe Research
{xubai, owang, caba, asagarwa}@adobe.com

Abstract

In this work we propose a new approach for accelerating the video editing process by identifying good moments in time to cut unedited videos. We first validate that there is indeed a consensus among human viewers about good and bad cut moments with a user study, and then formulate this problem as a classification task. In order to train for such a task, we propose a self-supervised scheme that only requires pre-existing edited videos for training, of which there is large and diverse data readily available. We then propose a contrastive learning framework to train a 3D ResNet model to predict good regions to cut. We validate our method with a second user study, which indicates that clips generated by our model are preferred over a number of baselines.

1. Introduction

Video editing is a time-consuming and challenging task traditionally performed by highly trained experts. In the most basic sense, video editing is time selection—selecting a series of clips that tell a story from raw, unedited footage, and then trimming each video down to its relevant part. As such, editors are performing two main tasks: the high-level task of deciding *which* content to show, and the low-level task of precisely placing cut points in a way that is not distracting to viewers. In this work we address the second task—the fine-scale placement of cuts (which can be equivalently thought of as clip trimming), assuming that the high-level direction of which clips to choose has been decided by the editor. We believe this component is better suited to automation as it is less dependent on high-level context or artistic choices, while being difficult and tedious to execute in practice, as it requires frame-level precision. Due to the increase in popularity of social video sharing websites, more and more novice users are creating and sharing edited video content, often times produced (shot, edited, and distributed) entirely on mobile devices. These users lack the time, expertise, and equipment to perform frame-level tasks such as cut placement.

[*]Work performed during internship at Adobe Research

In this work, we introduce the concept of "cut suitability", an instantaneous score for how good a cut would be if placed at that time. We ignore audio and focus purely on good visual times to cut. In our experience, audio and language determine clearly bad times to cut (e.g., during human voices, in the middle of sentences, and during loud noises), but otherwise provide a fairly uniform probability and is easy to determine using existing methods that can be combined with visual cuts in a late-fusion stage.

Before we begin to approach this problem of visual cutting, one might ask whether there is even any agreement among viewers as to what makes a "good" time to cut. We first validate this question by conducting a user study, which indicated that there is indeed a consensus about good and bad cut points. Generally good cut points occur at visually non-distracting times, in-between actions, or static moments right before or after camera motion, etc.

As cut suitability is a complex and hard to define function, we choose a data-driven approach that learns to associate visual features from real cut points. One challenge is that large scale datasets consisting of unedited and edited footage are hard to come by. In this work, we instead propose to use a weakly-supervised approach where we train a model entirely on edited video in the wild. This allows us to collect large-scale and diverse edited video (video with cuts), and then learn a one-sided function for the start and end placement of cuts (note that in this data, information *across* the cut for a single unedited clip is unavailable). We gather a dataset of 61,486 edited videos from YouTube and Vimeo.

Using this dataset, we propose a multimodal 3D Resnet architecture and train two separate models for starting and ending cut point prediction via contrastive learning. We design a progressive learning strategy to enforce the model to differentiate positive samples from negative samples with similar visual appearances. We use a randomized frame rate conversion method to augment the input videos, which effectively improves the model's robustness against video compression.

We then evaluate our models on both our collected dataset and a new set of unedited livestreaming videos. The experimental results show that our model is able to predict good cut positions close to the ground truth in the test set. Furthermore, we conduct a user study to evaluate the subjective preferences of human viewers.

Contributions.

We propose a model that predicts dense, continuous scores for cut suitability. Our key contributions include the following.

1. We introduce a novel task for computational video editing to automate the time-consuming manual process.
2. We propose a self-supervised contrastive learning framework that is completely data-driven and utilizes abundantly available edited videos without any manual annotation.
3. We define a number of baselines, and evaluate our approach with respect to human preference with a user study.

2. Related Work

Computational Cinematography. Previous papers on computational cinematography have looked towards automating difficult parts of the production process. Some work is on the camera side, such as stabilizing and centering the video on content as a post process [6]. In addition, prior work has investigated context-specific editing tasks, for example providing a transcript-centered interface for editing interviews [2], or a tool to leverages additional audio annotation created during filming [27]. Leake et al. [15] introduce a system to automatically generate edits from dialog scenes, where cuts are determined based on a set of high-level constraints, such as dialog and shot type. Arev et al. [1] propose a system to produce cuts using multiple social cameras. In this work, we propose a complementary video editing task: fine-scale placement of cuts for general-purpose footage, based on low-level content and motion cuts.

Edited Video in Computer Vision. Most video content online has at least a few basic edits, including cuts. This holds in genres ranging from short social clips to elaborate narrative videos. The computer vision community has leveraged such edited videos to train and benchmark diverse vision tasks including action recognition [13], speaker recognition[4], event localization[18], scene detection[20], among others. While edited video has served as a rich source to develop general-purpose video understanding systems, only a few approaches have leveraged the rich structure encoded within the video edits to learn video representations [19]. In this paper, we learn a cut suitability function that contrasts the visual features of moments *just* before and after a cut with features of all other moments in a video.

Video Shortening. The computer vision community has studied several video shortening problems, such as temporal action localization [17, 5] and event segmentation [23, 22]. While action boundaries could be places of high cut suitability, existing approaches for action localization have been designed to detect coarse moments in time, such as sport actions [11], high-level activities[3], and events in movies[18]. These are quite different movements to those who trigger cuts. Recently Shou *et al*. introduced the task of generic event boundary detection [22]. Even though their new study relaxes the shortening task from only actions to taxonomy-free event boundaries, it is unclear whether these general event boundaries correlate with good places to cut. Another widely-studied task addressing video shortening is video summarization [24, 7]. These methods process one video stream and cut that stream into multiple shots, making the video much shorter. Despite the similarities with our task, video summarization focuses on reducing the video length while maintaining the semantic meaning of the video unchanged. In short, all previous methods for video shortening offer only a rough time range prediction to delineate the start and end times of moments of interest in an untrimmed video. This observation makes them not viable as baselines for predicting frame-level cut suitability.

3. Problem Setting

As mentioned earlier, we are interested in the fine-scale localization of cuts. We formulate this task as two separate classification problems: whether a clip is a good starting clip, and whether a clip is a good ending clip. In each of these two problems, the task is then to predict a binary classification from the visual features contained in each clip, evaluated using a sliding window.

First, we verify whether there is in fact human agreement of "cut suitability" by conducting a user study. We developed a web interface that shows two clips, and ask the users which one is the better starting or ending clip. The user study shows that human's preference on which clip has a better cut agrees with the ground truth is 76% for starting point and 90% for ending point (elaborated in Table 2), indicating that there is agreement on what makes a good or bad cut. In this work, we use 2-second clips, which we found to contain enough semantic motion to establish context.

3.1. Learning from Edited Videos

Learning cut points could be done from paired data consisting of videos of raw footage, the trimmed clips and the timestamps of the edit points. However, acquiring diverse and large scale annotated data in this form is challenging.

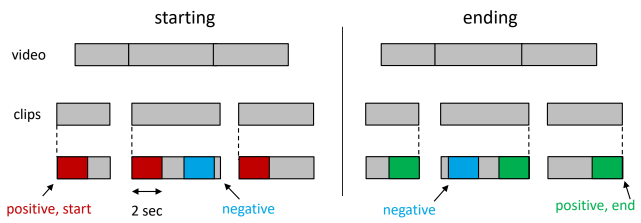

Figure 1. Data sampling for our method used in the start and end prediction tasks. Positive samples are clips that start (or end) with a cut, and negative samples drawn randomly from the rest of the video.

Alternatively, we look for edited videos (without original footage) that are widely available at scale on public video sharing platforms such as YouTube and Vimeo. We run a cut detection algorithm to identify cut points in edited footage [8] and break up the videos into clips. In this way we can collect cut points and the video content from one side of the raw footage (Fig. 1). Although the trimmed video content on the other side of each cut is missing, this setting has the advantage of being able to leverage virtually unlimited public videos for free.

We collect our dataset from public-video sharing websites by downloading random subsets from travel, narrative and food categories. In addition, we add the *vimeo-90k* dataset [29] to our collection. A sample of the collected video along with extracted clips is available online. [1] Our dataset contains 718 YouTube videos and 60,768 Vimeo videos. The total number of non-overlapping clips is 2.85 million. The average video duration is 205.86 seconds, and each video contains 46.51 clips on average.

4. Method

4.1. Architecture

3D CNN architectures have been proven to be useful for video-based tasks, such as action recognition [9]. We hypothesize that motion and object information are important factors in the determination of cut locations, and so we add additional inputs in the form of Mask R-CNN labels and optical flow. As shown in Fig. 2, each N-second video clip is therefore represented in 3 ways:

RGB pixel values. Frames are downsampled to 112×112, with 3-channel RGB format and 16 frames (8 fps). The tensor shape is $3 \times 16 \times 112 \times 112$.

Mask R-CNN labels. The pre-trained Mask R-CNN has 81 classes. We use a 1×1 convolution layer to project these features to 3-channels. The tensor shape is $3 \times 16 \times 112 \times 112$.

[1] http://www.yuzhonghuang.org/blog/VideoTrim/

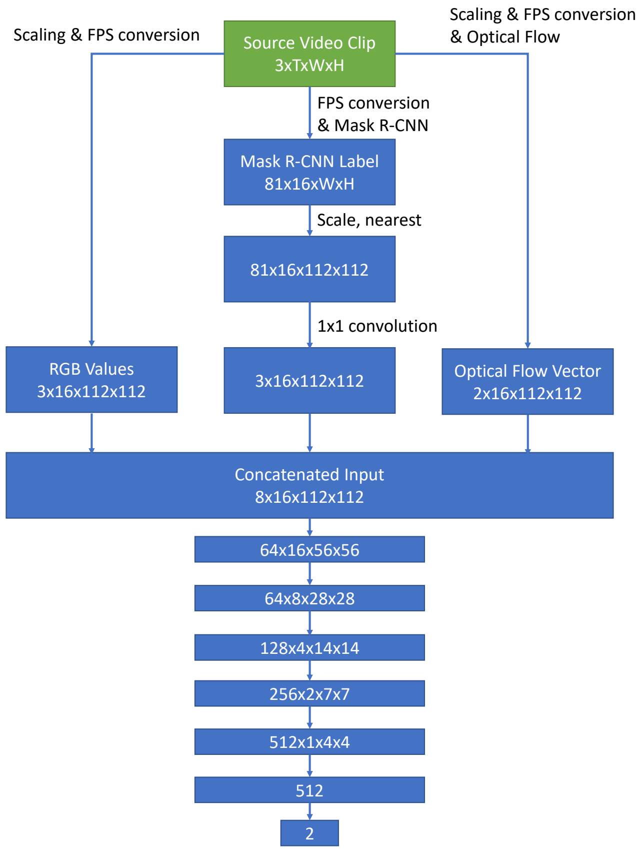

Figure 2. Architecture of proposed network. Please see Section 4.1 for details.

Optical Flow. We use a pretrained optical flow model [25] that computes a 2D motion vector for each pixel. The tensor shape is $2 \times 16 \times 112 \times 112$

These inputs are concatenated and passed through a series of 3D convolution layers. The model is trained to do binary classification. As described in Section 3, We train separate instances of this binary classifier for different tasks; predicting the *start* of a cut, and predicting the *end* of a cut. We also trained a unified model that performs 3-way classification, but it does not perform as well as the two separate models. We will elaborate this in Section 5.1.

We use the known cut locations as positive labels, and assign negative labels to randomly sampled N-second clips that are at least N seconds away from cut locations. Note that a random time is not necessarily a bad cut, so our negative samples are noisy. Considering a good cut point could last a few frames, we add a small degree of label smoothing here, which we found improved accuracy. For start (and similarly end) tasks, we assign soft label values $0.5, 0.25, 0.125$ to 3 frames after (before) the cut point. It is possible to train a model by minimizing the cross entropy between the model prediction and these smoothed labels, however we found that the trained model did not generalize well to test videos. (Section 5.1)

4.2. Contrastive Training

We observed two issues in network performance with the naive cross-entropy loss and simple frame rate conversion methods for the input video.

Model overfitting. We saw large gaps between the training and test accuracy in models with the cross-entropy loss, which might be caused by insufficient positive samples compared to the large network capacity. (See Baseline in Table 1)

Video compression. Most edited videos in our dataset have been compressed for streaming. Some compression algorithms use key-frames and motion estimation, which creates spatial and temporal dependencies in the processed frames. In particular, the frames near the edit points may be compressed differently, and the model may be trained to learn from the small structures or artifacts.

In order to address the first issue, we instead train our network using a supervised contrastive loss[14]. Given a mini-batch of $2N$ samples, $N_{\tilde{\boldsymbol{y}}_i}$ is the total number of samples in the mini-batch that has the same label, $\tilde{\boldsymbol{y}}_i$ as the anchor, i. $\boldsymbol{z}$ is the embedding of a sample. The supervised contrastive loss for this batch $\mathcal{L}^{sup}$ could be written as:

$$\mathcal{L}^{sup} = \sum_{i=1}^{2N} \mathcal{L}_i^{sup} \tag{1}$$

$$\mathcal{L}_i^{sup} = \frac{-1}{2N_{\tilde{\boldsymbol{y}}_i}-1}\sum_{j=1}^{2N}\mathbb{1}_{i\neq j}\cdot\mathbb{1}_{\tilde{\boldsymbol{y}}_i=\tilde{\boldsymbol{y}}_j}\cdot\log\frac{\exp\left(\boldsymbol{z}_i\cdot\boldsymbol{z}_j/\tau\right)}{\sum_{k=1}^{2N}\mathbb{1}_{i\neq k}\cdot\exp\left(\boldsymbol{z}_i\cdot\boldsymbol{z}_k/\tau\right)}$$

Several properties of the supervised contrastive loss ideally fit our task.

Contrastive power increases with more negatives [14] Videos are usually recorded at frame rates of least 24 fps. In a video clip, only a few frames are good cut points (positive), while most frames are not good places to cut (negative). So our task has inherently imbalanced labels, (each clip containing 1 positive and 34 negatives on average.) and contrastive loss could take advantage of it.

Non-symmetric loss for positive and negative samples As seen in Eq. 4.2, z_j and z_k are a positive and negative samples respectively. $\mathcal{L}^{sup}$ is directly correlated with $z_i \cdot z_j$ while correlated to the reciprocal of $z_i \cdot z_k$. The loss term for negative sample will quickly decay as $z_i \cdot z_k$ increase, but won't decay for positive samples.

This is a desired property for our task. Such setup of loss will enforce positive samples to get similar embeddings with other positive samples, even if they have very different visual appearances. For negative samples, we design it to

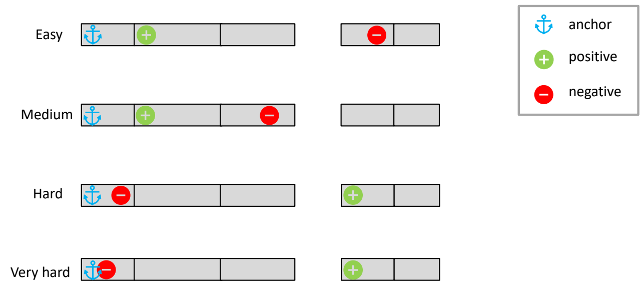

Figure 3. Sampling Schemes. In each row we show two edited videos (grey bars), with cuts shown as vertical lines. By changing where positive and negative samples are drawn from relative to an anchor placed on a cut in the first video, we can make the task easier, or harder.

enforce them to be far away from positive samples. The intuition behind this is that there is only a small subset of clips that are indeed good cut points with shared consensus, while the negative clips are much more diverse.

In addition, we employ a curriculum learning strategy, where we progressively generate samples with increasing levels of difficulty for the network to learn. Fig. 3 shows an example of the schemes for the starting clip task. A positive sample from a clip is used as the anchor. In *easy* scheme, we sample the positive from another clip in the same video, and the negative from a different video. In *medium* scheme, the negative comes from a different clip in the same video, which is more similar to the anchor. We then make the scheme harder by moving the negative sample in the same clip as the anchor, and moving the positive sample to a different video. In this way, the network needs to learn shared features between two visually distinct positive samples, and learn to distinguish samples of different classes from the same scene. It becomes even more challenging by moving the negative sample very close to the anchor and forcing the network to differentiate two samples that have very similar visual appearances, as shown in the *very-hard* scheme.

During training, we start from the *easy* sampling scheme for better convergence, and gradually shift towards more difficult sampling schemes to improve classification accuracy on challenging videos.

The contrastive loss and the sampling schemes encourages the network to cluster inter-class samples and distinguish intra-class samples regardless of their visual similarity. As a result, we can train a much more robust model with high accuracy (Section 5.1).

4.3. Temporal Augmentation

To address the second issue, we propose a new temporal augmentation method.

We convert the input videos to a fixed FPS of 8 such that every input video sample contains 16 frames. FFmpeg

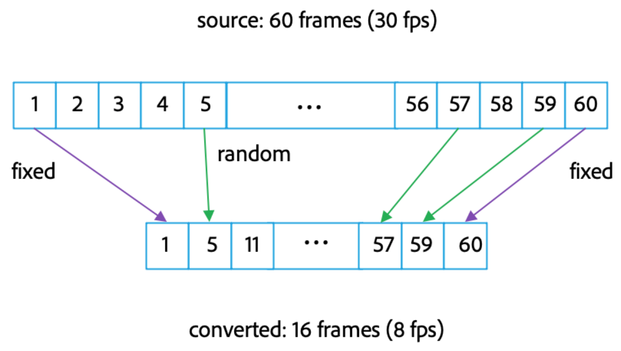

Figure 4. Randomized Frame Rate Conversion. We augment our training samples by jittering sampling locations during rate conversion.

[26] supports several frame conversion methods, including frame rounding, frame blending, and optical flow based blending, but these methods are designed for good viewing experiences rather than effective data sampling methods. Frame rounding cannot utilize available data effectively as it rounds to nearest frames to the desired timestamps. Frame blending and optical flow methods both produce different frames than the source thus the model might learn from data in different domains.

Inspired by [16, 28], we propose a randomized frame sampling method that selects the first and last frame of the source video, and then randomly selects the remaining 14 frames in ascending order (Fig. 4).

This randomized frame sampling method augments the training data with more variants. It prevents the network from learning any temporal patterns, structures or artifacts from video compression, and as a result we found that models trained with this augmentation generalized better to unseen videos.

Our model was trained with 0.1 learning rate using SGD optimizer with momentum, with an early stopping strategy that stops when the validation loss did not decrease for 3 epochs. We then recovered the lowest validation loss weight. The model was trained for 78 epochs. Training took about 42.7 hours on a single Nvidia V100.

5. Experiments

Baselines Recent works related to video cut point prediction include highlight prediction, video segmentation and action recognition. To the best of our knowledge, there are no models specifically designed to predict dense, continuous scores of cut feasibility at frame level. Therefore, we introduce a number of naive baselines by ablating the various components of our method, and also experiment with using an action recognition model [12] as our baseline, with the hypothesis that cut suitability is often a function of when actions have been completed. We use this baseline by fixing the pretrained action recognition weights, and fine-tuning the last layer on our collected dataset using a standard cross-entropy loss.

5.1. Classification Evaluation

As ground truth labels in unedited footage are not widely available, we use our collected YouTube and Vimeo videos for self supervision and quantitative evaluation. Please note that although we use edited videos for evaluation, our model only sees continuous (unedited) video clips, whereas the shotcut detection model sees edits. (Fig. 5). Our model needs to learn from the video content from a single clip and infer whether it is a good start or end.

We evaluate on a 80/20 split of training and testing, and report training and test accuracy on both start and end tasks in Table 1. From these results, we conclude

1. The model generalizes well to a very large set of diverse, unseen videos (87.51% test accuracy for start task and 79.84% for end task). This suggests that our model is able to learn the common features at the start or end of human-edited clips.

2. Cut point prediction is a high-level task that benefits high-level features (semantic labels, motion vectors) derived from training related tasks. The ablation study results show that our proposed model improves the classification accuracy of the baseline model by 28.54% for start task and 21.99% for end task.

3. The start task and end task have different behavior and presumably are learning different features. The 3-way classification model has lower accuracy than separate models in start and end tasks. Based on this experiment result, we use two models instead of a unified model.

4. The contrastive loss and temporal augmentation significantly increase the model's prediction accuracy. In particularly, the gaps between training and test are decreased, which suggests better robustness and generalization.

5.2. Distribution of model prediction scores

As shown in Fig. 6, we compute the average scores of all test videos on 16 frames after a true starting point. These 16 frames are all the temporally downsampled frames within the 2-second clip. We can see that the predicted score is the highest on the first frame and quickly drops to small values on frames away from the start. This distribution confirms that the model's prediction approximates the ground truth. We observe a similar distribution for the ending task model where the high values are concentrated towards the last frame.

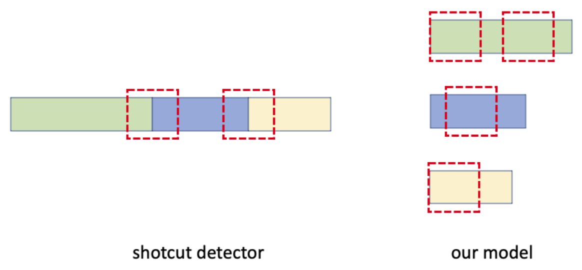

Figure 5. Difference between the shotcut detector and our model. The shotcut detector fires when the window includes a cut. Our model sees continuous, unedited clips and the window does not contain any cuts.

Input	Model	Start Task Train	Start Task Test	End Task Train	End Task Test
RGB	XE (Baseline)	91.71	58.97	95.69	58.85
RGB	3 way XE	54.79	55.05	58.24	55.37
RGB	CT w/ TA	77.12	68.33	70.07	61.14
RGB	CT w/o TA	85.23	58.44	82.45	57.14
Optical Flow [25]	XE	75.23	62.89	69.33	57.14
Mask RCNN [10]	XE	62.15	56.85	61.24	53.73
RGB+Optical Flow	XE	89.28	75.91	88.93	68.29
RGB+Mask RCNN	XE	87.88	78.00	86.78	78.42
RGB+Optical Flow+Mask RCNN	XE	**97.45**	68.29	**94.53**	65.32
⋆ RGB+Optical Flow+Mask RCNN	CT w/ TA	93.42	**87.51**	89.76	**79.84**

Table 1. Quantitative comparison of accuracy values for different baselines and ablations. In this table XE stands for cross entropy loss, CT stands for contrastive loss, and TA stands for temporal augmentation. The last row prefixed by ⋆ is our proposed model. We can see that adding the higher-level features improves accuracy, and our proposed contrastive learning training scheme improves generalization to our test set.

5.3. User Study Evaluation

We conduct a user study to analyze the users' subjective evaluations on the predicted scores of cut feasibility. The participants are 15 individuals hired from the video production community on upwork.com who worked on video projects ranging from consumer to professional levels. We present to the participants a web page that shows a pair of short clips, and ask them to review and click on the one with better starting or ending, respectively; left-right order is randomized. We generate a total of 3,000 pairs of clips that are sampled from the test set and a new, out of domain set of unedited livestreaming videos (all natural videos) from twitch.com. To add redundancy, we design the user study so that each web page is viewed and clicked by five different users, which allows us to analyze the consensus of each selection among different individuals.

There are 3 types of tasks when users select the clip:

1. *Start*. Choose the clip with better starting from two N-second-clips.

2. *End*. Choose the clip with better ending from two N-second-clips.

3. *Clip*. Choose the clip with better starting and ending. The clips are generally longer (between 2 seconds to 15 seconds) than those in the other two tasks, and they are closer approximation to real world video editing tasks.

Dataset \ Task	Start	End	Clip
GT	90.00	76.66	81.66
Test Set	80.45	69.44	71.66
Livestream	69.16	63.33	66.66

Table 2. User study results. Average evaluation scores (in percentage) for different tasks and datasets. A higher number indicates that human viewers agree more with the model's prediction or the ground truth.

For each video pair, we generate the positive one by sampling local peak values from the continuous curve predicted by the model. A peak is defined as a local maximum inside the clip with predicted value above 0.9 threshold. The peak value indicates that the video sample is likely to have a good starting/ending cut point. Similarly, we generate the negative one by sampling local valley values below 0.1. If the users are able to select the positive clip, it suggests that the model is making a correct classification that matches human viewers' subjective decision for what makes a "good cut location".

To find an upper bound of human agreement, and to validate our ground truth, we also ask the users to perform the same set of tasks on clips generated with ground truth labels, by sampling the true clip boundaries as positive, and random samples elsewhere as negative.

We compute the subjective score of each selection by majority voting, which means we say that one clip "wins" if there are more than half of the users making that decision. The average accuracy for three tasks are reported in Table 2. The results show that an expected upper bound for human agreement should be around 77%-90%, as demonstrated on the *GT* dataset where cuts have been hand chosen by editors. This indicates that there is a statistically significant consensus among users that the positive and negative samples can be separated by humans. Further, we see that on our held out test set, humans agree with our model 69%-80% of the time. We also evaluate 0-shot dataset transfer by testing on the aforementioned out-of-domain *livestream* dataset (unedited livesteaming video), and see that accuracy scores are lower 63%-69%, although still are significantly over chance (50%). This is likely due to the different domain than the training data.

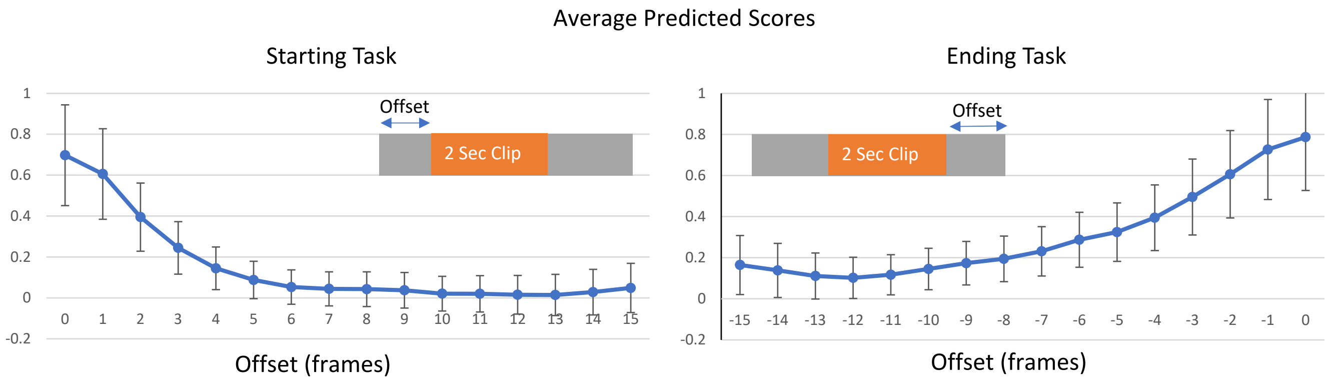

Figure 6. Average predicted scores on frames near the clip boundary. The x-axis is the offset of the input clip as it shifts away from the boundary. The model's prediction is highly concentrated on the true positive at clip boundaries.

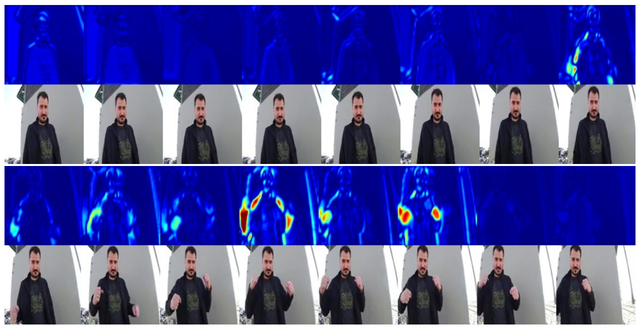

Figure 7. Grad-CAM visualization of an input video clip. We can see that the network pays attention to regions with motion, e.g., the waiving hands.

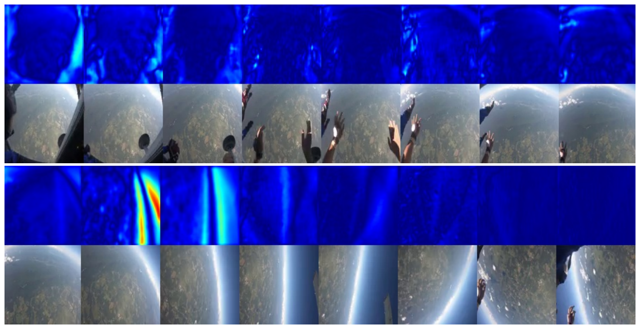

Figure 8. Grad-CAM visualization of an input video clip. We can see that the network pays particular attention to salient features such as the horizon

5.4. Grad-CAM Visualization

To understand what our model is looking at, we utilize Grad-CAM [21] to visualize the activation heat map. A few examples are shown in Fig. 7, 8. Our model detects semantically meaningful regions in the scenes and pays attention to moving objects, landmarks and human faces, etc. This analysis gives us insights into the model's decision on evaluating a good cut position.

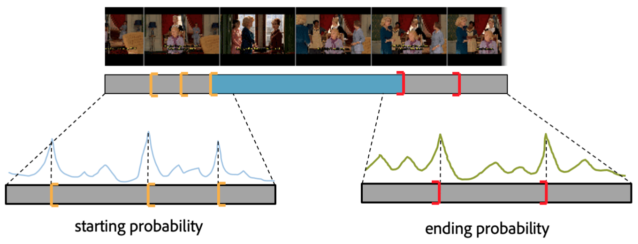

Figure 9. An example of automatic cut point prediction using our model. Given an input video, the model predicts two curves. The starting cuts (orange) and ending cuts (red) are detected at peak values of the curves. A candidate clip (blue segment) can be generated by combining the cut points in either automatic or interactive fashion.

6. Applications

We use the models to predict the cut suitability on every frame. The local maximums (or values above a threshold) can be used as candidate cut points for starting or ending tasks. A starting cut point can be combined with an ending cut point to generate a clip. We envision possible applications to assist users with video editing tasks such as cutting, trimming and selection. See Fig. 9.

Cut point suggestion. The model recommends the most likely cut positions that can be combined to produce clips.

Refine clip boundaries of a rough selection. The user selects a clip and then the clip boundaries will snap to the closest candidate cut points predicted by the model.

Avoid bad cuts. Cutting is disabled in regions with very low prediction scores.

7. Conclusion and Future Work

In conclusion, we take a first step towards evaluating and learning from the cutting points, introduce a new problem of cut suitability prediction from video data using weakly-supervised edited videos collected in-the-wild, and demonstrate potential applications of the proposed method. Our paper provides practical examples of how AI could be used for understanding and accelerating video editing.

Using the proposed method, we are able to simplify the video editing process by "snapping" cuts to frames that are preferred by viewers, as validated by a user study. Our method uses contrastive learning with a sampling curriculum designed for this task that improves results over a number of baselines. In addition, we have showed that adding features derived from other high-level tasks such as motion estimation and object segmentation improve the overall accuracy of our approach.

We hope that as edited videos become a more common form of communication and self-expression, that this and other similar computational cinematography technologies will enable new users to participate in this medium. In the future, we would like to investigate the decision-making process of humans when editing videos, which will help us understand the gap between human preference and the model's prediction. We plan to add the audio cut points in a late-fusion stage for regions of videos containing speech or music. While we report the average accuracy on a very large dataset (2.85 million clips), we can augment the study by classifying the videos into different categories using metadata or scene classifiers, and report the numbers for each video type such as sports, touring, speech, etc. Training specialized models for each video domain will help increase the accuracy.

References

[1] Ido Arev, Hyun Soo Park, Yaser Sheikh, Jessica Hodgins, and Ariel Shamir. Automatic editing of footage from multiple social cameras. *ACM Trans. Graph.*, 33(4), July 2014. 2

[2] Floraine Berthouzoz, Wilmot Li, and Maneesh Agrawala. Tools for placing cuts and transitions in interview video. *ACM Transactions on Graphics (TOG)*, 31(4):1–8, 2012. 2

[3] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015. 2

[4] Mark Everingham, Josef Sivic, and Andrew Zisserman. Hello! my name is... buffy"–automatic naming of characters in tv video. In *BMVC*, volume 2, page 6, 2006. 2

[5] Adrien Gaidon, Zaid Harchaoui, and Cordelia Schmid. Temporal localization of actions with actoms. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2782–2795, 2013. 2

[6] Michael L Gleicher and Feng Liu. Re-cinematography: Improving the camerawork of casual video. *ACM transactions on multimedia computing, communications, and applications (TOMM)*, 5(1):1–28, 2008. 2

[7] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in neural information processing systems*, pages 2069–2077, 2014. 2

[8] Michael Gygli. Ridiculously fast shot boundary detection with fully convolutional neural networks. *CoRR*, abs/1705.08214, 2017. 3

[9] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6546–6555, 2018. 3

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. 6

[11] Haroon Idrees, Amir R Zamir, Yu-Gang Jiang, Alex Gorban, Ivan Laptev, Rahul Sukthankar, and Mubarak Shah. The thumos challenge on action recognition for videos "in the wild". *Computer Vision and Image Understanding*, 155:1–23, 2017. 2

[12] Hirokatsu Kataoka, Tenga Wakamiya, Kensho Hara, and Yutaka Satoh. Would mega-scale datasets further enhance spatiotemporal 3d cnns?, 2020. 5

[13] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2

[14] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning, 2020. 4

[15] Mackenzie Leake, Abe Davis, Anh Truong, and Maneesh Agrawala. Computational video editing for dialogue-driven scenes. *ACM Trans. Graph.*, 36(4):130–1, 2017. 2

[16] Juhyun Lee. Non-deterministic video frame sampling to thwart frame insertion attacks. 2017. 5

[17] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *European Conference on Computer Vision*, 2018. 2

[18] Xiaolong Liu, Yao Hu, Song Bai, Fei Ding, Xiang Bai, and Philip HS Torr. Multi-shot tempo-

ral event localization: a benchmark. *arXiv preprint arXiv:2012.09434*, 2020. 2

[19] Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Human mesh recovery from multiple shots. *arXiv preprint arXiv:2012.xxxxxx*, 2020. 2

[20] Anyi Rao, Linning Xu, Yu Xiong, Guodong Xu, Qingqiu Huang, Bolei Zhou, and Dahua Lin. A local-to-global approach to multi-modal movie scene segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10146–10155, 2020. 2

[21] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017. 7

[22] Mike Zheng Shou, Deepti Ghadiyaram, Weiyao Wang, and Matt Feiszli. Generic event boundary detection: A benchmark for event segmentation, 2021. 2

[23] H. S. Sokeh, V. Argyriou, D. Monekosso, and P. Remagnino. Superframes, a temporal video segmentation. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 566–571, 2018. 2

[24] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. Tvsum: Summarizing web videos using titles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5179–5187, 2015. 2

[25] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. 2018. 3, 6

[26] Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006(146):10, 2006. 5

[27] Anh Truong, Floraine Berthouzoz, Wilmot Li, and Maneesh Agrawala. Quickcut: An interactive tool for editing narrated video. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, page 497–507, New York, NY, USA, 2016. Association for Computing Machinery. 2

[28] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 5

[29] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision (IJCV)*, 127(8):1106–1125, 2019. 3