— Supplementary Material —
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

Here, we present additional details, results and analyses that could not be included in the main paper due to page-limit constraints. All references and figures in this supplementary file are self-contained.

1 Anatomy of Video Editing (AVE): Dataset

In Fig. 1, Fig. 2 and Fig. 3, we plot the class-wise distribution statistics for all shot attributes. We can infer two key things from the figures. First, there is a long tail label distribution problem in many of the shot attributes as discussed in the main paper. Second, we conduct all of our experiments on a very balanced training and evaluation sets. Beyond the shot-level attributes, we further analyze transition patterns between contiguous shots in `AVE`. In Fig. 4, we plot the most frequent shot attribute transitions with respect to their probability of occurrence. Please refer to the Project page for further qualitative analyses on the proposed dataset and benchmark tasks.

2 Experimental Results and Discussion

2.1 Experimental Settings

Network Architecture. We use `ResNet-101` [4] and `R-3D` [16] as visual backbone networks for image and video inputs, respectively. For feature extraction, we remove the last `fc` layer from both networks, and obtain a visual feature of size 1024 and 512 for `ResNet-101` and `R-3D` backbones, respectively. Visual backbone networks are initialized with pretrained weights (`ResNet-101` - pretrained on ImageNet [3] and `R-3D` - pretrained on Kinetics-400 [5]) and fine-tuned during training. To extract features from the audio input, we designed `AudioNet`, which is a feed-forward network with three `convolutional` and two `linear` layers. We use `kernel` sizes of $\{(7\times3),(5\times3),(5\times5)\}$ and `stride` sizes of $\{(3\times1),(3\times1),(3\times3)\}$ for the 3 `convolutional` layers, respectively. Each `convolutional` layer is followed by a `ReLU` activation layer. After processing the audio input through the `convolutional` layers, we apply a `global pooling` layer to obtain a one dimensional audio feature. This feature is further processed using 2 `linear` layers with a `ReLU` activation layer in between. The visual

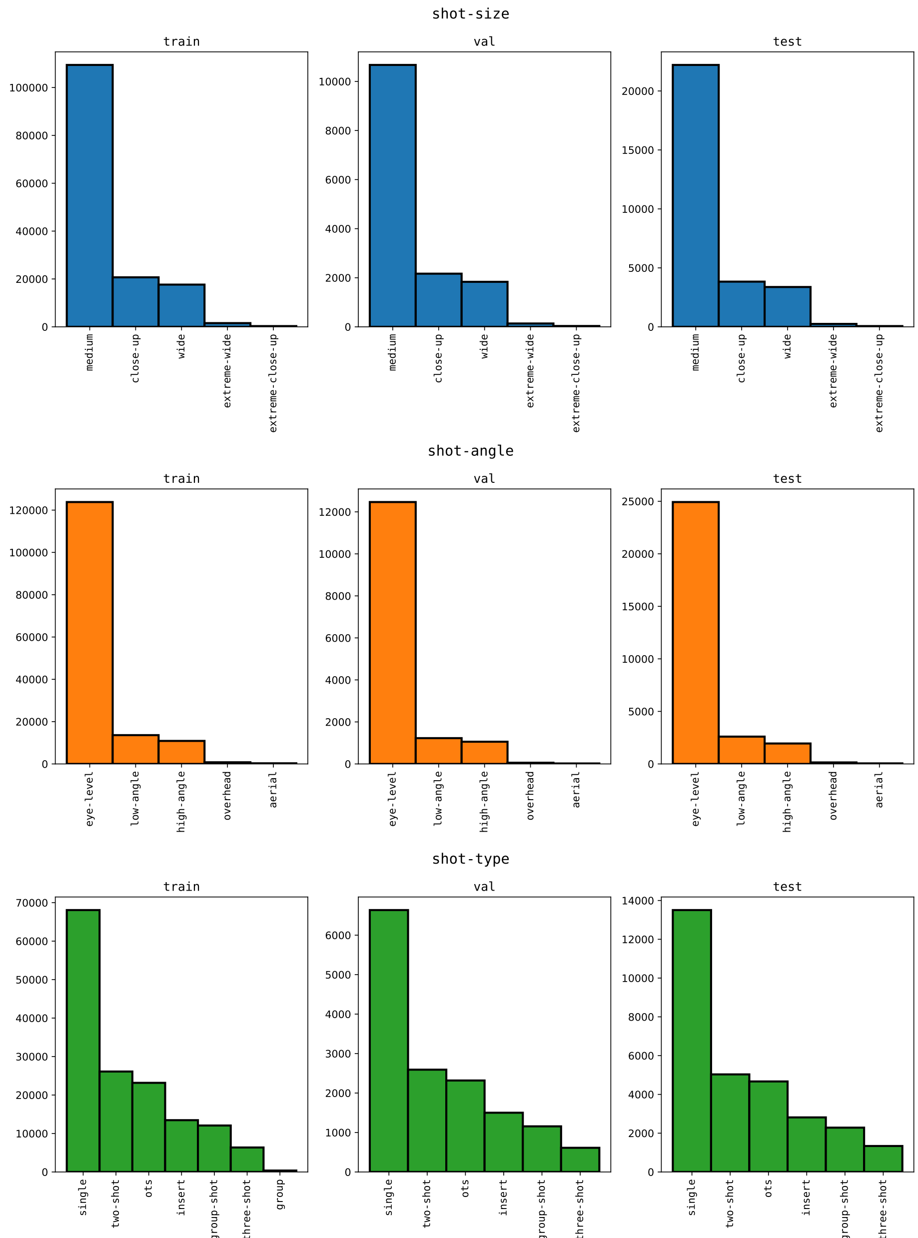

Fig. 1: Class-wise distribution statistics in training, validation and testing splits for `shot size`, `shot angle` and `shot type` attributes.

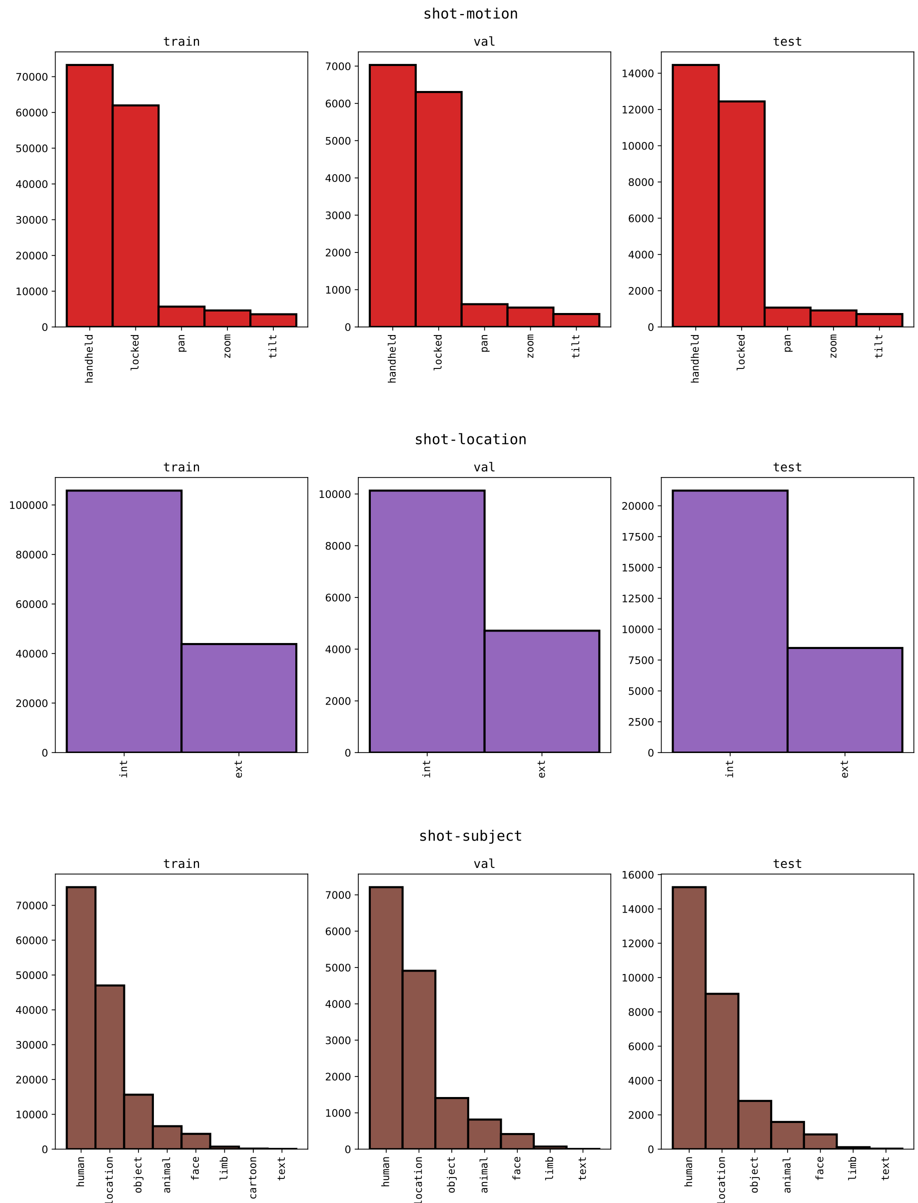

Fig. 2: Class-wise distribution statistics in training, validation and testing splits for `shot motion`, `shot location` and `shot subject` attributes.

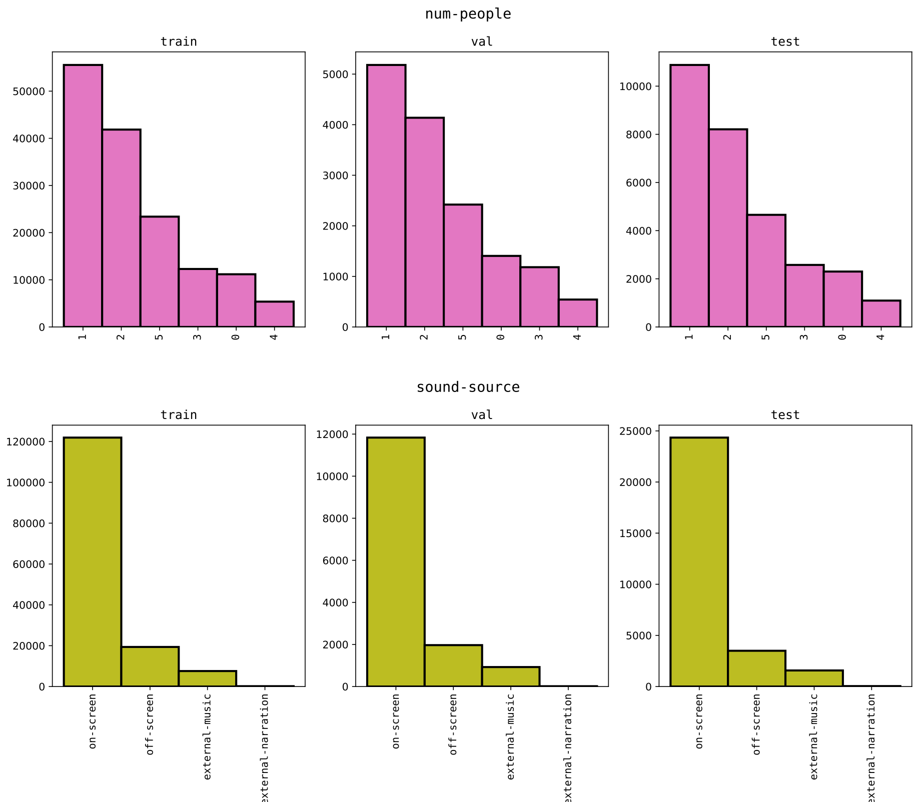

Fig. 3: Class-wise distribution statistics in training, validation and testing splits for `num. of people` and `sound source` attributes.

and audio features are then concatenated channel-wise to obtain an audio-visual feature. We use the same visual and audio networks for all tasks in the main paper.

For *shot attributes classification* task, the audio-visual feature is feed into eight classifier networks. We use a network with two `linear` layers as a classifier network for each attribute. A `dropout` and a `ReLU` activation layers are used in between the `linear` layers. The final `linear` layer outputs a logit vector with size equal to the number of classes for the respective shot attribute.

For *shot sequence ordering* task, the `feature fusion` network in the late feature fusion baseline (`Baseline-I`) is a simple network with `linear`, `ReLU` activation and `dropout` layers. The `feature fusion` network inputs the hierarchically concatenated features and outputs a fused feature representation. The classifier networks in both `Baseline-I` and `Baseline-II` (early input fusion) have the same architecture as the classifier in the previous task.

For *next shot selection* task, a recurrent network is used to learn the shot sequence pattern. We use a two-layered (stacked) `LSTM` module with a hidden

Table 1: Train-val-test split statistics for different tasks

	# of shots	Train	Val	Test	Total
`Num. of scenes`	-	3914	559	1118	5591
`Shot attributes classification`	1	151053	15040	30083	196176
`Camera setup clustering`	-	-	-	-	-
`Shot sequence ordering`	3	75000	7500	15000	97500
`Next shot selection`	9	75000	7500	15000	97500
`Missing shot attributes prediction`	3	75000	7500	15000	97500

size of 128 to obtain `anchor`, `positive` and `negative` embeddings from an input sequence of audio-visual features.

For *missing shot attributes prediction* task, a `label-to-feature (L2F)` network is used to incorporate the attributes of the input shots along with their audio-visual features. We use a simple 1-layered linear network to transform an attribute vector of size 8 to a feature representation. A `feature fusion` network (like the one used in shot sequence ordering task) is then used to combine the cross-modal features extracted from the two input shots. The output of the `feature fusion` network is feed into eight classifiers to predict the attributes of the missing shot. We use the same classifier architecture as the one used in shot attributes classification task.

Dataset. We follow a `train-val-test` scene split of 70-10-20 in all experiments (see Table 1). As the scenes in the proposed dataset are non-overlapping, the train, validation and test splits are disjoint sets. For *shot attributes classification* task, we use all the shots in the respective scene split for training and evaluation, *i.e.* 151053 training, 15040 validation and 30083 testing shots. 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. As shown in Table 1, we create a total of 97500 shot triplets, where 75000, 7500 and 15000 shot sequences are used for training, validation and testing, respectively. The sequences are randomly shuffled during training for *shot ordering* task, thereby creating an augmented dataset that is significantly larger than the initial samples. For next shot selection task, we sample 9 consecutive shots from a scene 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. We generate 75000 training, 7500 validation and 15000 testing shot sequences for conducting experiments.

Implementation Details. We use `ffmpeg` to extract frames of size 1280×720 from a given shot clip. We crop 50 pixels from the top and bottom corners of each frame to remove the logo of the channel from where the movie scenes are crawled and create consistency across the dataset [3]. We then uniformly sample 16 frames and resize each sampled frame to a size of 320×130 to represent a shot clip as a `video` input to `R-3D`. We pick the central frame from the extracted frame sequence of a shot and resize it to 640×260 to represent a shot clip as a single `frame` input to `ResNet-101`. The audio file from a given shot clip is

[3] MovieClips YouTube Channel

extracted in a '`.wav`' format using `ffmpeg`. We then use the `torchaudio` [17] library to load the '`.wav`' file as a 2D `spectrogram` image of size 513×32, by applying zero padding when necessary. The spectrogram image is then feed into `AudioNet` to extract audio features. We implement our models in PyTorch [12]. We use SGD optimizer [14] with a *momentum*, *weight decay* and initial *learning rate*, 0.9, $1e-4$ and $1e-3$, respectively, for all tasks.

For *shot attributes classification* task, we train our framework for 100 epochs, with the learning rate decaying by 0.1 at 40, 60 and 80 epochs. We use a batch size of 50 during training. We deal with the long tail label distribution problem by adjusting the logits [10] of each classifier according to the label frequencies in the respective shot attribute. Note that logit adjustment is used only during training. To scale the cross entropy loss from each classifier for multi-task training, we follow [7] and implement dynamic weight averaging technique with a temperature parameter $T = 2$. For single-task training, we simply optimize the cross entropy loss of the classifier.

For *camera setup clustering* task, we experiment with several feature extraction methods. For `SIFT` [9], we use the implementation from `OpenCV` [4]. For `CLIP` [13], we use the image encoder part of the official pretrained model with '`ViT-B/32`' backbone. For `ResNet-101` [4] and `R-3D` [16], we use the pretrained models (with the last `fc` layer removed) from PyTorch [12]. For standard clustering algorithms such as `K-Means` [8], Hierarchical Agglomerative Clustering (`HAC`) [11] and `OPTICS` [1], we use the `scikit-learn` implementations [5]. For `FINCH` [15], we use the official code.

For *shot sequence ordering* task, we train both `Baseline-I` and `Baseline-II` for 100 epochs, with the learning rate decaying by 0.1 at 40, 60 and 80 epochs. We used a batch size of 50 during training. We use the cross entropy loss for training both baselines.

For *next shot selection* task, we train our network for 200 epochs, with the learning rate decaying by 0.1 at 100, 150 and 175 epochs. We used a batch size of 100 during training. We use the supervised NT-Xent loss [2,6] with a temperature parameter of 0.02 for training.

For *missing shot attributes prediction* task, we use the same training setting as the shot attributes classification task.

2.2 Experimental Results

Shot Attributes Classification. In Table 2, we compare the class-wise performance in each attribute for a network trained *with* and *without* taking the long tail label distribution problem into account. Here, we consider a multi-task training setting with `video + audio` input. As can be seen from Table 2, for attributes with imbalanced label distributions such as `shot size` and `shot angle`, naiv̈ely trained network performs very well for the dominant classes but extremely poorly for low frequency classes. On the other hand, a network trained

[4] OpenCV SIFT
[5] Scikit-learn Clustering

Table 2: Class-wise performance analysis on shot attributes classification.

	Multi-task training (Video + Audio)				
		Naïve training		Logit adjustment	
Attribute		Val	Test	Val	Test
Shot size	Medium	92.7	93.0	58.0	56.0
	Wide	69.4	66.2	54.7	55.0
	Close-up	19.0	22.1	67.1	65.9
	Extreme-wide	0.0	0.0	77.5	82.8
	Extreme-close-up	0.0	0.0	61.5	65.4
	Average	36.2	36.4	66.8	65.0
Shot angle	Eye-level	98.1	97.6	62.3	61.7
	High-angle	27.5	26.6	45.3	48.2
	Low-angle	12.6	14.0	49.3	54.1
	Overhead	0.0	0.0	43.3	18.1
	Aerial	0.0	0.0	92.9	65.4
	Average	27.6	27.7	58.6	49.5
Shot type	Single	84.6	85.9	68.1	70.7
	Group-shot	60.7	63.5	64.1	65.1
	Two-shot	57.9	55.3	52.3	49.7
	Insert	64.2	65.3	76.7	78.9
	OTS	68.8	70.9	75.5	76.4
	Three-shot	22.4	25.2	45.6	50.9
	Average	59.8	61.0	63.7	65.3
Shot motion	Locked	79.9	79.8	82.0	82.1
	Handheld	81.5	79.3	27.4	26.3
	Tilt	0.0	0.0	40.9	38.7
	Zoom	0.0	0.0	31.7	22.0
	Pan	0.0	0.0	41.2	47.2
	Average	32.3	31.8	44.6	43.2
Shot location	Ext	73.2	67.8	84.1	81.7
	Int	92.8	94.0	83.3	85.7
	Average	83.0	80.9	83.7	83.7
Shot subject	Human	94.9	95.3	96.0	81.2
	Face	93.9	94.1	90.0	77.2
	Animal	55.0	47.3	74.8	71.3
	Object	30.6	36.6	40.3	45.6
	Location	6.0	4.5	19.3	17.6
	Limb	0.0	0.0	31.2	34.0
	Text	0.0	0.0	0.0	0.0
	Average	40.0	39.7	50.2	46.7
Num. of people	0	77.0	74.7	90.7	88.3
	1	81.9	82.9	73.5	76.0
	2	72.0	72.3	62.3	62.7
	3	25.7	27.0	30.9	34.2
	4	0.0	0.1	46.1	47.8
	5	74.0	73.3	62.1	58.5
	Average	55.1	55.3	60.9	61.4
Sound source	On-screen	100.0	100.0	56.7	56.3
	Off-screen	0.0	0.0	19.9	18.7
	External-music	0.0	0.0	41.1	46.3
	External-narration	0.0	0.0	46.1	34.3
	Average	25.0	25.0	41.0	38.9
Average		44.9	44.7	**58.7**	**56.7**

with logit adjustment gives a relatively balanced per-class accuracy, and hence a better overall performance.

In Table 3, we summarize the results of using different input representations for shot attributes classification task in a multi-task training setting. It can be

Table 3: Quantitative analysis on shot attributes classification.

	Multi-task training											
	`Frame`				`Video`				`Video + Audio`			
	Naiv̈e		Logit adj.		Naiv̈e		Logit adj.		Naiv̈e		Logit adj.	
Attribute	`Val`	`Test`	`Val`	`Test`	`Val`	`Test`	`Val`	`Test`	`Val`	`Test`	`Val`	`Test`
`Shot size`	38.1	37.9	62.0	66.8	35.7	35.5	67.8	66.9	36.2	36.4	66.8	65.0
`Shot angle`	32.7	32.6	63.9	55.9	25.8	25.8	62.2	53.2	27.6	27.7	58.6	49.5
`Shot type`	62.0	63.2	64.3	64.7	59.5	60.8	63.9	64.9	59.8	61.0	63.7	65.3
`Shot motion`	26.4	26.0	31.7	33.5	32.1	31.7	42.8	42.7	32.3	31.8	44.6	43.2
`Shot location`	82.6	80.0	84.0	82.1	82.9	81.9	84.4	83.3	83.0	80.9	83.7	83.7
`Shot subject`	42.4	41.2	51.0	46.8	40.0	39.8	50.8	47.4	40.0	39.7	50.2	46.7
`Num. of people`	57.9	56.8	61.6	60.2	55.0	55.1	61.3	61.2	55.1	55.3	60.9	61.4
`Sound source`	25.0	25.0	31.3	32.0	25.0	25.0	34.4	32.6	25.0	25.0	41.0	38.9
`Average`	45.9	45.3	**56.2**	**55.2**	44.5	44.4	**58.4**	**56.5**	44.9	44.7	**58.7**	**56.7**

inferred from Table 3 that using a single `frame` to represent a shot generally results in a lower performance compared to using `video` and `video + audio`. However, it is worth noticing that, for attributes such as `shot size` and `shot angle` which, in essence, does not require temporal or audio information, using a `frame` representation outperforms other types of inputs. On the other hand, for attributes such as `shot motion` and `sound source` which are closely associated with temporal and audio contexts, respectively, using only `frame` as an input gives a significantly worse performance.

Missing Shot Attributes Prediction. In Table 4, we present the results on four shot attributes, *i.e.* `shot location`, `shot subject`, `num. of people` and `sound source`, for a model trained in a multi-task setting. This is in continuation of the results from Table 7 in the main paper. As can be seen from Table 4, the proposed model outperforms the naiv̈e `dominant label` prediction baseline by a large margin. It can also be inferred that incorporating the attributes of the input shots along with other representations consistently improves model accuracy across all attributes.

As shown in Table 5, the multi-task training setup leads to an unbalanced performance when using `frame` as an input, *i.e.* the performance gap between `shot size` and other attributes is notably large in comparison with using other input representations (refer to Table 7 in the main paper). This is mainly because the model overfitted to the `shot size` attribute for this particular input setup. To verify this hypothesis, we train our model in a single-task setting for

Table 4: Quantitative analysis on missing shot attributes prediction.

	Shot location		**Shot subject**		**Num. of people**		**Sound source**	
Method	`Val`	`Test`	`Val`	`Test`	`Val`	`Test`	`Val`	`Test`
`Dominant label`	50.0	50.0	14.3	14.3	16.7	16.7	25.0	25.0
`Frame`	73.0	70.7	27.1	23.0	31.3	26.8	31.2	31.6
`Frame + Attributes`	94.6	94.4	43.4	43.1	37.9	37.4	34.8	37.1
`Video`	83.8	82.2	30.3	27.4	36.6	35.4	28.8	32.6
`Video + Attributes`	93.9	92.8	44.4	43.9	38.4	37.4	35.7	33.8
`Video + Audio`	85.8	83.4	31.0	28.7	37.3	36.3	29.7	33.3
`Video + Audio + Attributes`	**95.0**	**94.4**	**45.5**	**45.0**	**38.9**	**38.4**	**40.3**	**41.2**

Table 5: Multi-task vs. single-task analysis on missing shot attributes prediction.

Setting	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
Multi-task	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
Single-task	Frame	29.4	32.2	25.6	28.4	26.5	25.2	27.2	27.7
	Frame + Attributes	34.3	36.6	30.3	32.9	36.0	34.6	29.4	30.2

Table 6: Additional results for shot attributes classification task

	Frame				Audio + Video					
	CLIP		**ResNet-101**		**Weight**		**Freeze**		**Baseline**	
Attribute	Val	Test	Val	Test	Val	Test	Val	Test	Val	Test
Shot size	52.9	51.3	62.0	66.8	66.9	66.0	51.9	50.4	66.8	65.0
Shot angle	54.3	54.9	63.9	55.9	59.0	50.2	54.2	54.3	58.6	49.5
Shot type	56.0	58.0	64.3	64.7	62.4	64.6	49.4	51.2	63.7	65.3
Shot motion	36.6	37.6	31.7	33.5	44.7	43.2	35.6	35.2	44.6	43.2
Shot location	82.1	81.0	84.0	82.1	83.8	83.7	84.6	85.0	83.7	83.7
Shot subject	45.2	42.9	51.0	46.8	48.8	45.8	47.6	43.9	50.2	46.7
Num. of people	56.9	57.2	61.6	60.2	60.0	61.4	53.3	52.6	60.9	61.4
Sound source	44.1	43.4	31.3	32.0	43.1	40.1	41.9	40.9	41.0	38.9
Average	53.5	53.1	**56.2**	**55.2**	58.6	**56.9**	52.3	51.7	**58.7**	56.7

each attribute. As can be inferred from Table 5, the single-task setting gives a relatively balanced performance across attributes.

2.3 Discussion

Here, we discuss the concerns raised by anonymous reviewers regarding shot-attributes classification task. The additional results that addressed the concerns are presented in Table 6. Experiments are conducted in a multi-task setting applying the logit adjustment [10] technique.

Why not use SOTA methods such as CLIP? We experimented with using pre-trained CLIP's visual encoder as a backbone network for shot attributes classification task, however, we observed an inferior performance compared to using ResNet-101 (see **CLIP** column in Table 6).

Why use vector concatenation for feature fusion? Because it is simple. We also experimented with weighted combination of visual and audio features as a fusion mechanism, however, we did not observe any significant improvement in performance (see **Weight** column in Table 6)

Did you try freezing the backbones? We did. However, we opted for fine-tuning the backbone networks along with the classifiers because it resulted in a much better performance (see **Freeze** column in Table 6).

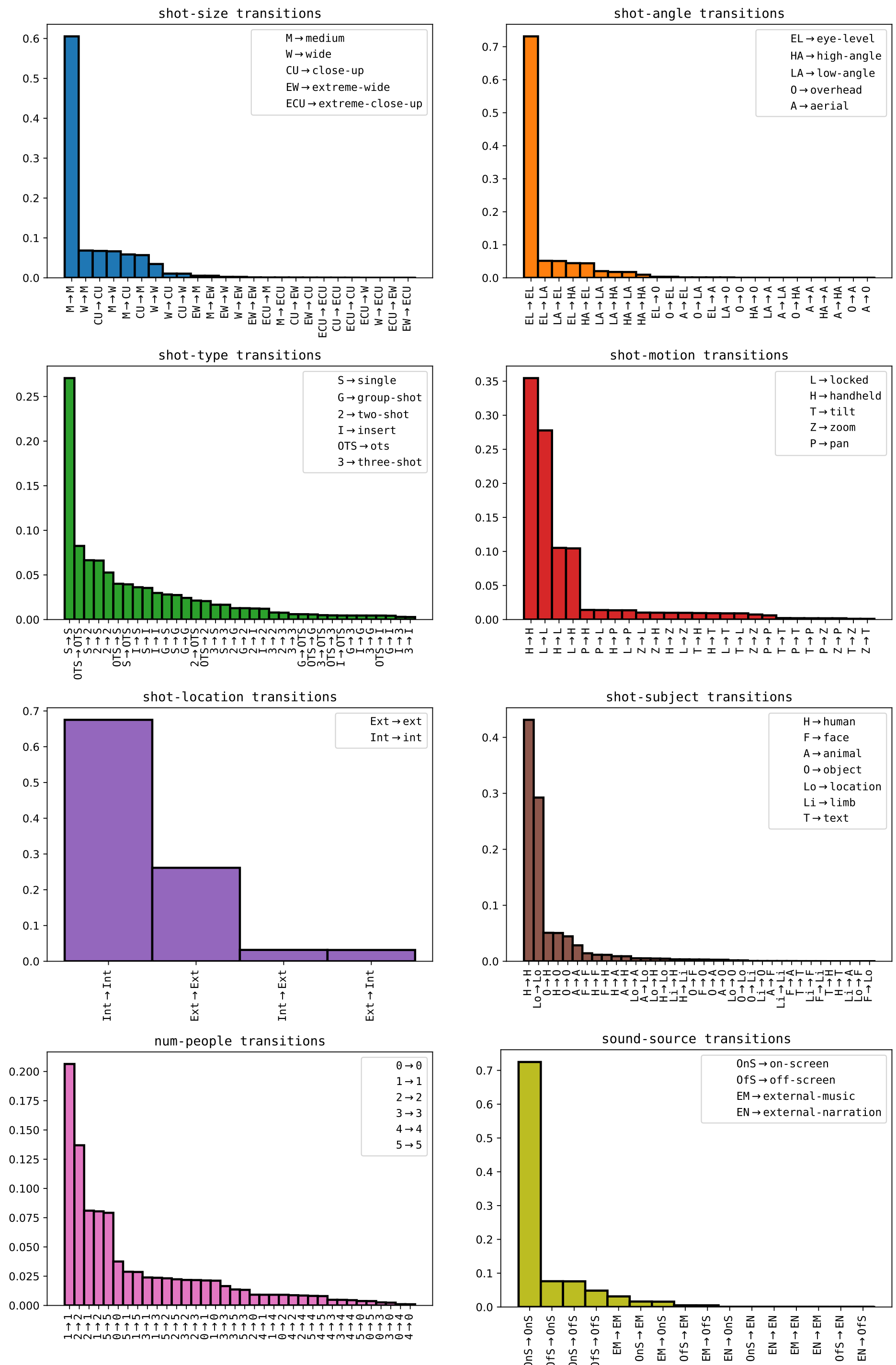

Fig. 4: Most frequent shot attribute transitions in `AVE`. The `y-axis` indicates the probability of occurrence and the `x-axis` denotes the transition.

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) 6
2. 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) 6
3. 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) 1
4. 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) 1, 6
5. 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) 1
6. 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 6
7. 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
8. Lloyd, S.: Least squares quantization in pcm. IEEE transactions on information theory **28**(2), 129–137 (1982) 6
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004) 6
10. 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, 9
11. Müllner, D.: Modern hierarchical, agglomerative clustering algorithms. arXiv preprint arXiv:1109.2378 (2011) 6
12. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019) 6
13. 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) 6
14. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016) 6
15. 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) 6
16. 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) 1, 6

17. Yang, Y.Y., Hira, M., Ni, Z., Chourdia, A., Astafurov, A., Chen, C., Yeh, C.F., Puhrsch, C., Pollack, D., Genzel, D., Greenberg, D., Yang, E.Z., Lian, J., Mahadeokar, J., Hwang, J., Chen, J., Goldsborough, P., Roy, P., Narenthiran, S., Watanabe, S., Chintala, S., Quenneville-Bélair, V., Shi, Y.: Torchaudio: Building blocks for audio and speech processing. arXiv preprint arXiv:2110.15018 (2021) 6