

APPENDIX

A MORE IMPLEMENTATION DETAILS

A.1 DETAILED MODEL ARCHITECTURE

Fig. 1 illustrates the detailed components in our model architecture, which is briefly represented in Fig. 2. HiSum has an encoder for each semantic level, which helps to learn more appropriate representation at each level. The architecture of the encoder principally follows Dosovitskiy et al. (2021), but we do not utilize dropout (Srivastava et al., 2014) in MLP module of the encoder, since we observe slightly under-performing result with it. The encoder E is fully presented in Fig. 1a. Here, ℓ means the number of stacked Transformer blocks. After each level obtains the representing feature from its encoder, a regressor R estimates scores at the corresponding level. HiSum has the regressor for each semantic level as with the encoder. The input of the regressor is the feature mentioned above and it is illustrated in Fig. 1b. Final output of the regressor is the importance score at each semantic level.

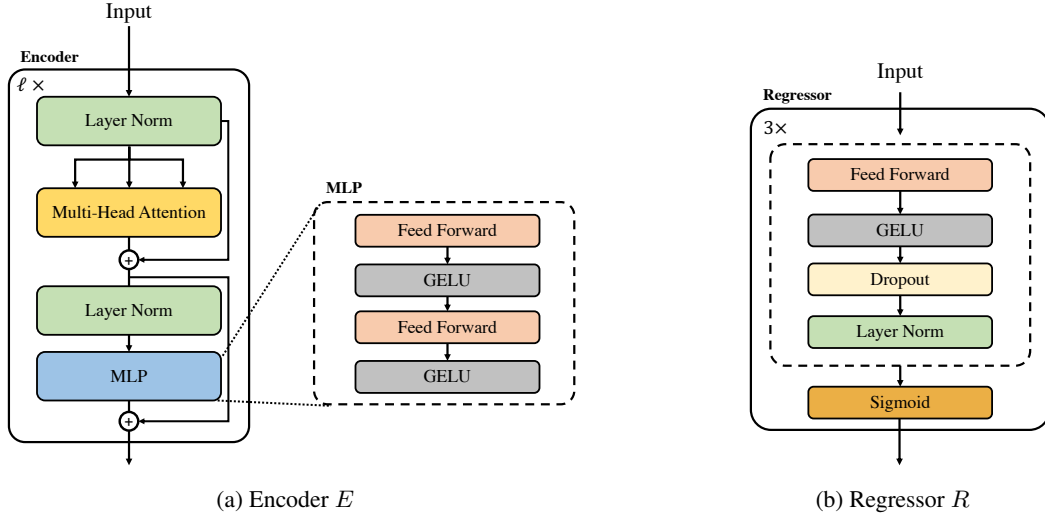


Figure 1: Detailed HiSum model components. (a) The encoder E follows vision transformer structure. (b) The regressor R is composed of stacked feed forward networks.

A.2 MORE ON HYPERPARAMETERS

We describe more detailed hyperparameter for training HiSum. In the last column of Table 1, we reduce the dimension of the first input feature to $d_2 = 512$ (SumMe) and $d_2 = 640$ (TVSum) through multiple fully-connected layers. For the encoder, we set $\ell = 1$ in every level for SumMe and $\ell = 2$ in every level for TVSum. We use the multi-head attention layer with 8 heads. The hidden dimension of MLP is set to $4 \times d_1$, which is 4096. For the regressor, the hidden dimension of each fully-connected layer is set to $[1024, 512, 256, 1]$, followed by a sigmoid activation.

Based on extensive experiments with regard to λ_i , we set $\lambda_1 = 1.3$, $\lambda_2 = 0.4$ and $\lambda_3 = 1.3$ on SumMe and $\lambda_1 = 0.75$, $\lambda_2 = 1.5$ and $\lambda_3 = 0.75$ on TVSum, chosen by cross-validation.

A.3 GENERIC EVENT BOUNDARY DETECTION FOR VIDEO SUMMARIZATION

We use generic event boundary detection model to select event-level boundaries. UBoCo (Kang et al., 2022) was originally evaluated on Kinetics-GEBD (Shou et al., 2021), a standard benchmark dataset for event boundary detection. Kinetics-GEBD videos have similar length and are much shorter than the videos in summarization dataset. Kang et al. (2022) was able to extract fixed number of frames from the videos for this reason. However, summarization dataset consists with various

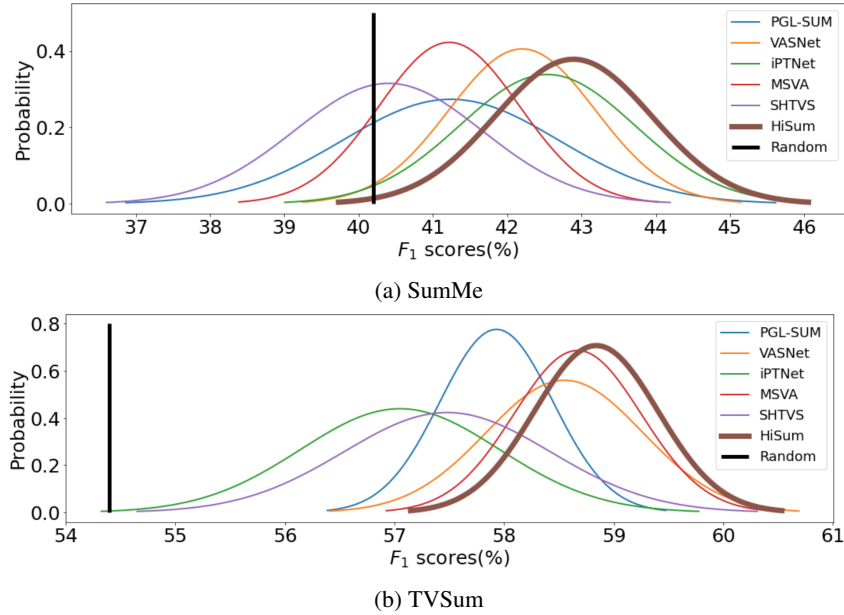


Figure II: Performance comparison considering standard deviation. (a) F_1 scores on SumMe. (b) F_1 scores on TVSum.

length of videos and they are much longer than the Kinetics-GEBD videos, which is not a situation to extract fixed number of frames from the videos.

Therefore, we modify the original UBoCo to adapt to our summarization problem. Among the four encoders in UBoCo, Mixer (Guo et al., 2022) has token-mixing MLP which requires fixed number of tokens in an image. For the case of videos, however, the number of frames is obviously not a constant. We thus remove the Mixer and change it into a raw visual feature path and make TSM from it. Also, there is an important hyperparameter called “gap” which predefines interest region of the similarity. We change this hyperparameter into ratio out of the total length. Likewise, other hyperparameters that have dependency with video length are all changed to ratio-based.

B STATISTICAL SIGNIFICANCE IN SUMMARIZATION TASK

Due to the lack of data, performance of the summarization models fluctuate a lot depending on the initial model parameter values. In order to derive statistically significant results, we run every model five times per split and come up with the average performance reported in Table 1. Fig. II shows the results in normal distribution form, where the mean and standard deviation is computed from the 25 experiments (5 test videos times 5 repetition with different random seeds). From Fig. IIa on the SumMe dataset, PGL-SUM (Apostolidis et al., 2021b) and MSVA (Feng et al., 2018) have large standard deviation and low average F_1 score so that their probability of showing lower performance than random summary is substantially high. Fig. IIb on the TVSum dataset, iPTNet (Jiang & Mu, 2022) shows low performance and large standard deviation. Considering both benchmark datasets, VASNet (Fajtl et al., 2018) and our HiSum model show reliable results with acceptable small standard deviation.

C DIVERSITY AND IMPORTANCE DATASETS

This section explains detailed information on how we create Diversity-Balanced (DB), Diversity-Unbalanced (DU), and Importance-Unbalanced (IU) sets. All datasets are made by concatenating clips from videos in the original dataset. Original dataset is organized by Zhang et al. (2016a). Zhang et al. (2016a) extracted GoogleNet features from the dataset and also unite format of summaries of TVSum and SumMe. We use this format of the dataset as a baseline. First, We split videos into 3:1:1

for train, validation, test set and also maintain the 5-fold cross validation scheme. Within each set, three clips are concatenated for the diversity (DB, DU) sets and four videos for the importance set (IU). Corresponding summaries are also concatenated but summaries are meaningful when whole video context is considered. Therefore, when concatenating videos, we always make sure at least one clip is used in full-length. For the DB set, all the clips have similar length. For DU set, clips other than full-length clips are 0.3 lengths of full-length clip. They all have similar summary ratio which is summary length over clip length. There are multiple raters for each video, so summary ratio is substituted as the average of frame importance score in each clips. For the IU set, one clip is used in full-length of original video and other short clips, randomly sampled from three different videos have 1/10 length of the long one.

D QUALITATIVE RESULTS

D.1 SEMANTIC HIERARCHY AS VIDEO STRUCTURE

Fig. III, Fig. IV, and Fig. V illustrate examples of the constructed semantic hierarchy for a few videos from the SumMe dataset. Through these examples, we can see how semantic hierarchy describes the video structure and how semantic boundaries split semantic units appropriately.

D.2 MACHINE GENERATED SUMMARIES ON DB, DU, IU

Fig. VI, Fig. VII, and Fig. VIII show a few examples of the generated summaries on DB, DU and IU datasets using our trained HiSum. The yellow lines represent video boundaries where the video is concatenated.

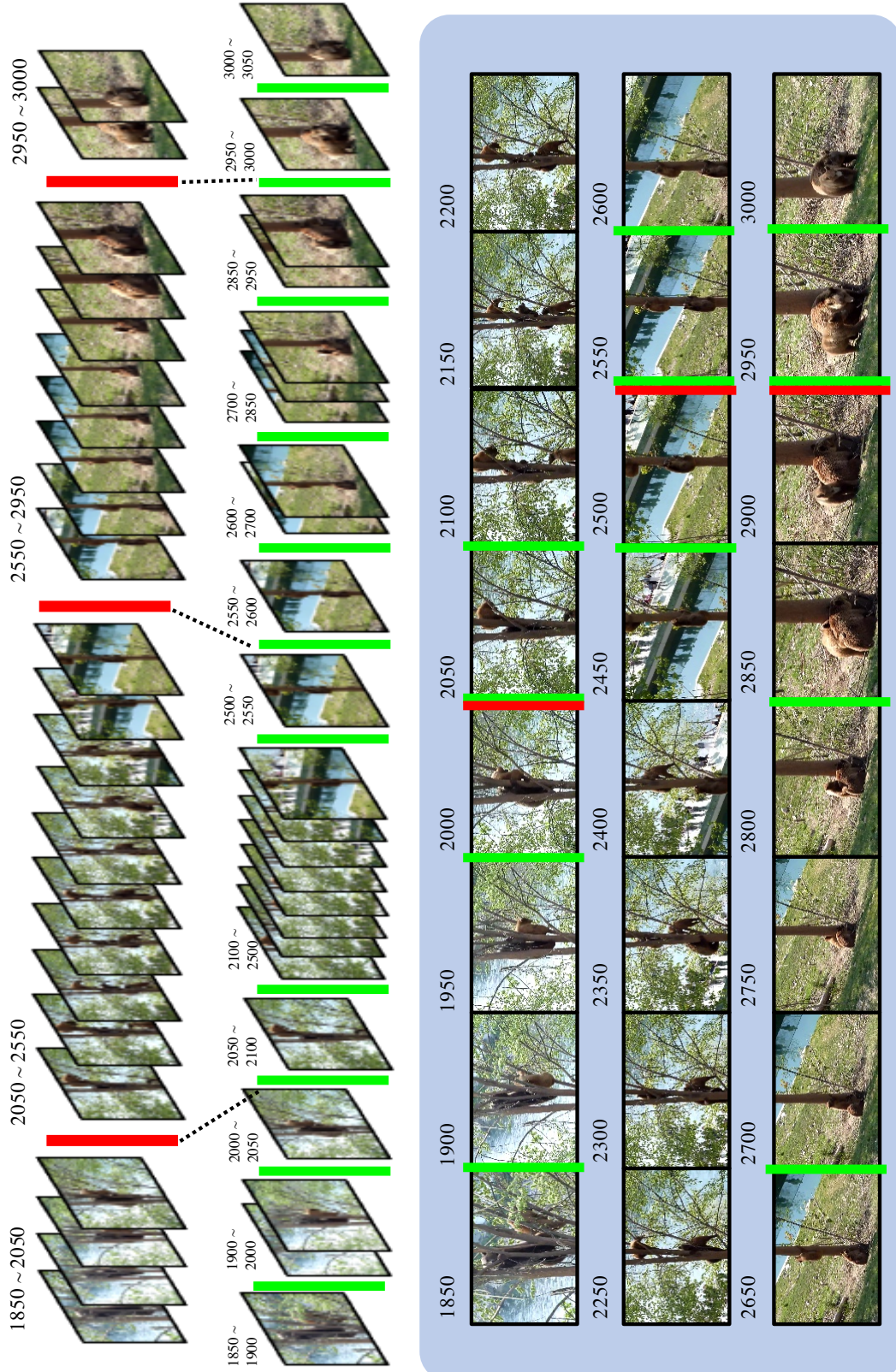


Figure III: Discovered semantic hierarchy of “Bearpark_climbing” video from SumMe.

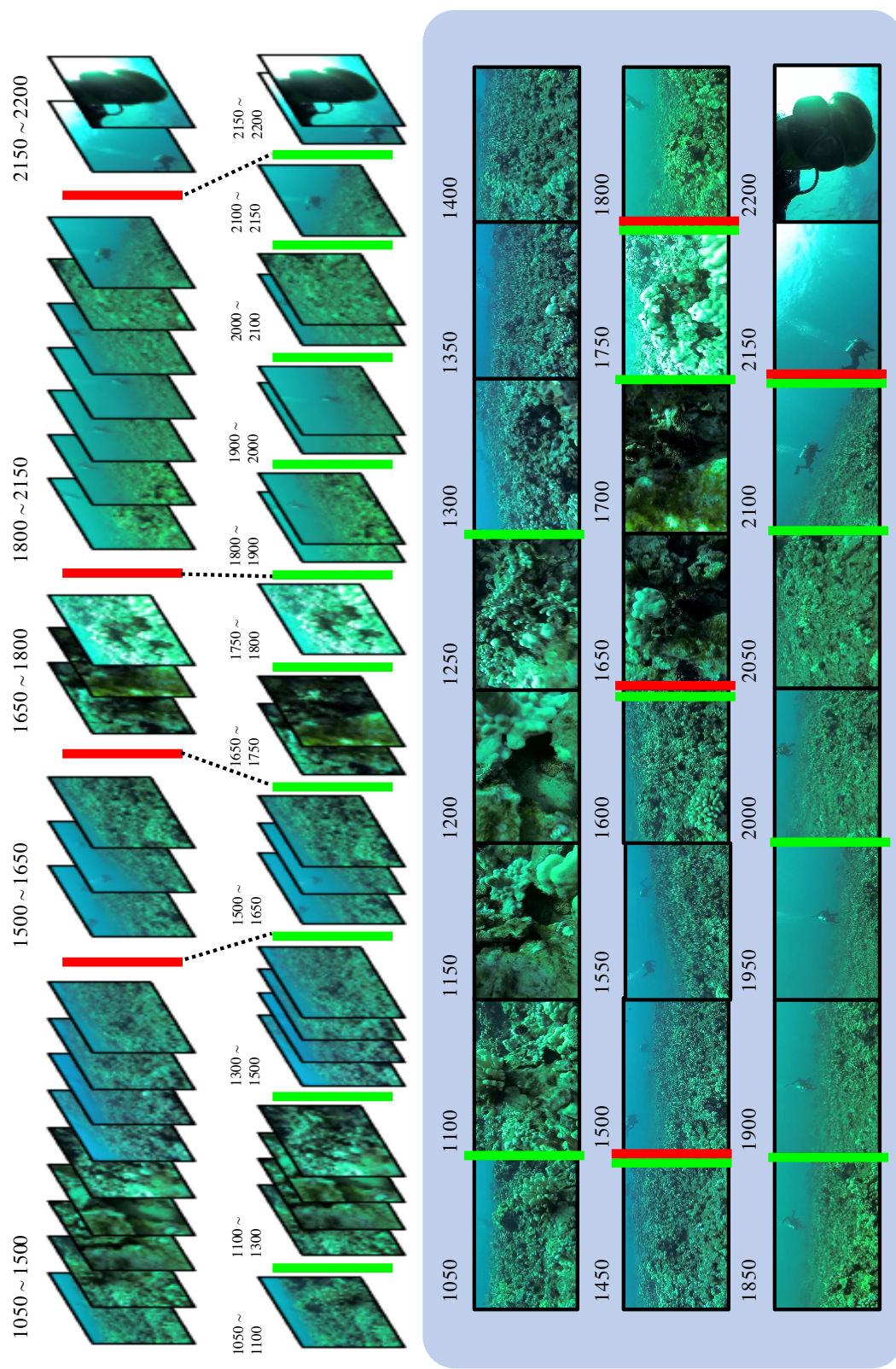


Figure IV: Discovered semantic hierarchy of "Scuba" video from SumMe.

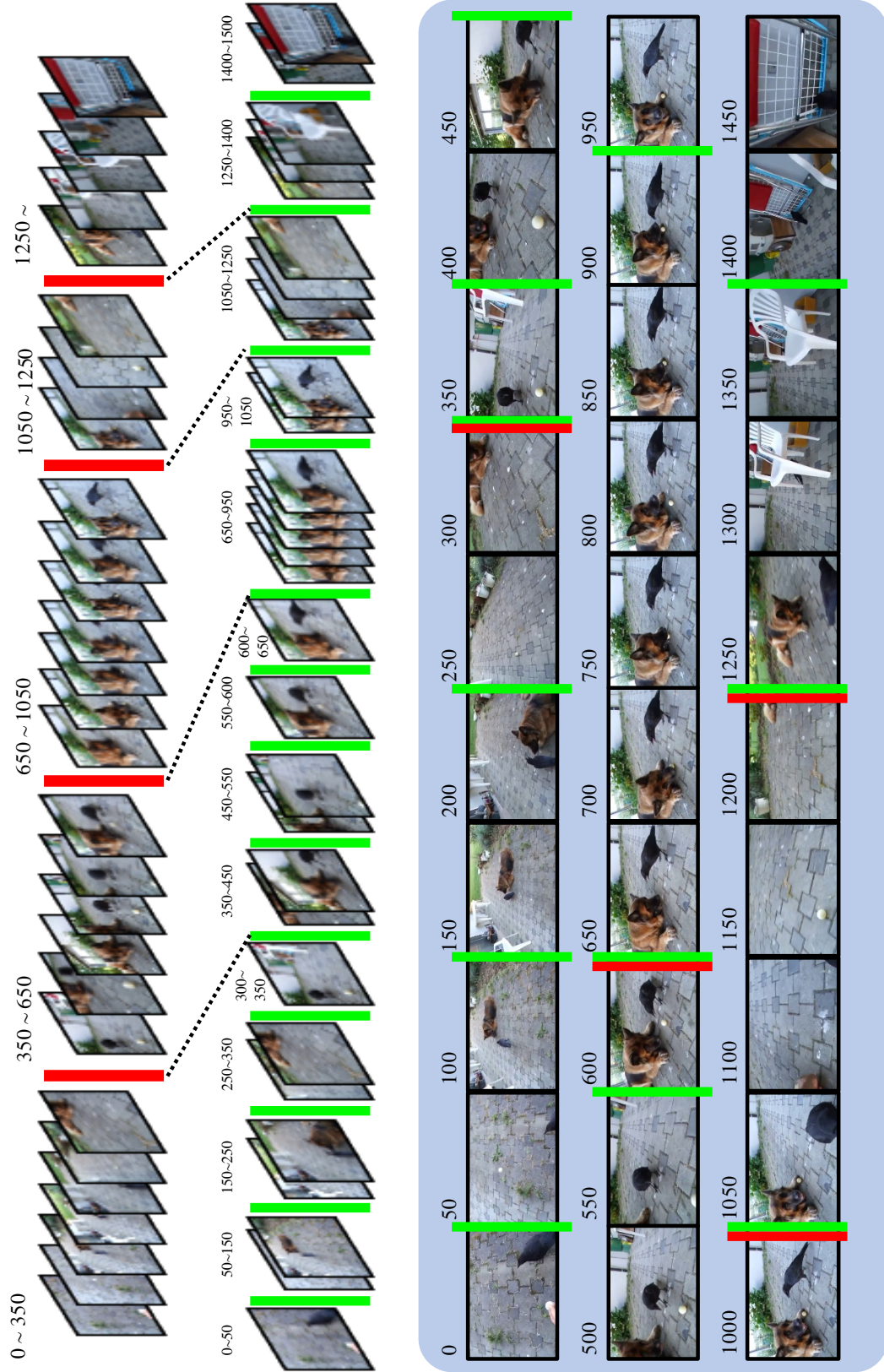


Figure V: Discovered semantic hierarchy of "Playingball" video from SumMe.

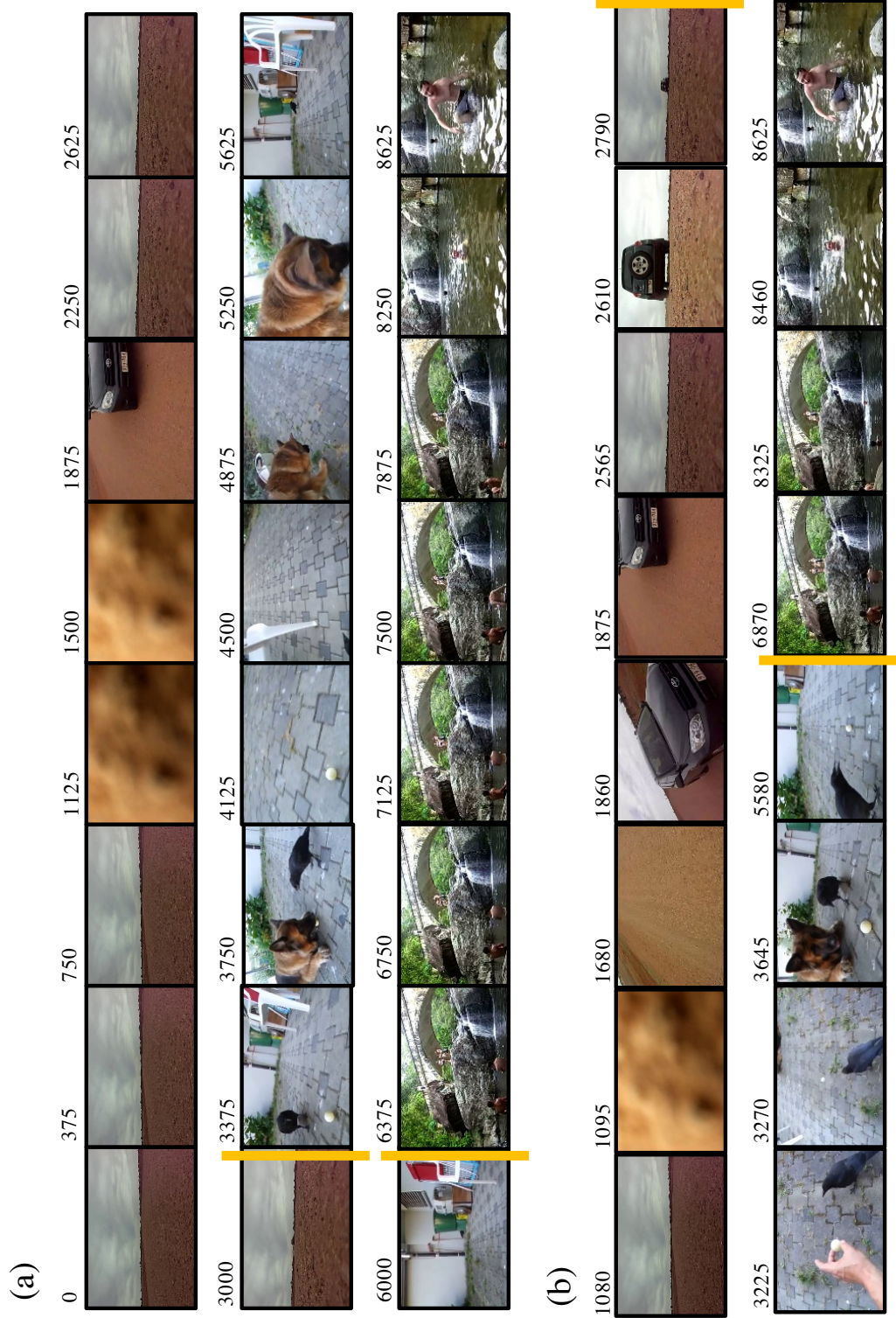


Figure VI: Machine generated summary of video25 in the DB dataset. (a) The entire sequence of the video. (b) Generated summary by our trained model. Our summary assigns similar weights between the concatenated clips, as expected.

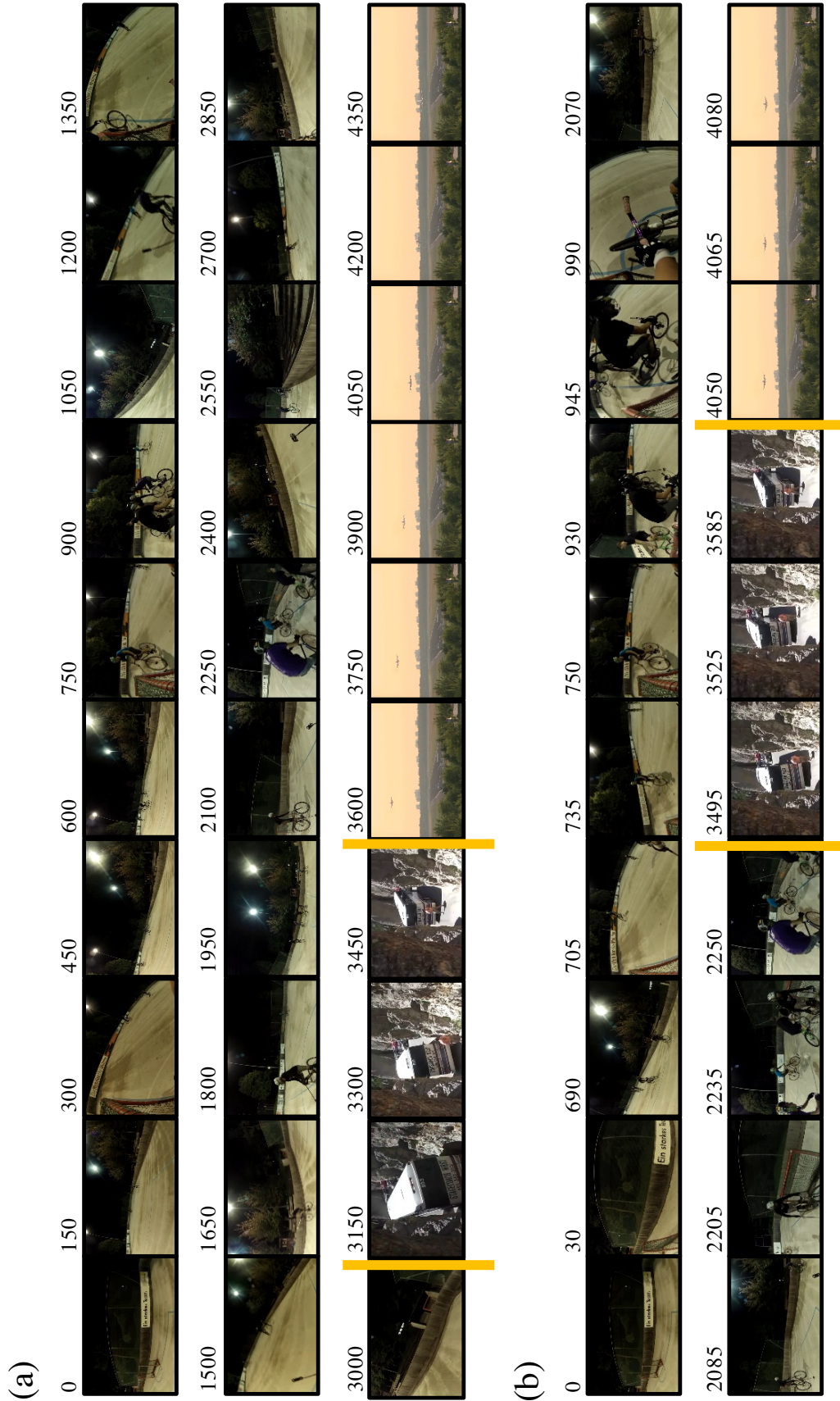


Figure VII: Machine generated summary of video4 in the DU dataset. (a) The entire sequence of the video. (b) Generated summary by our trained model. Although clips in DU have different lengths, all of them are included in a balanced manner.

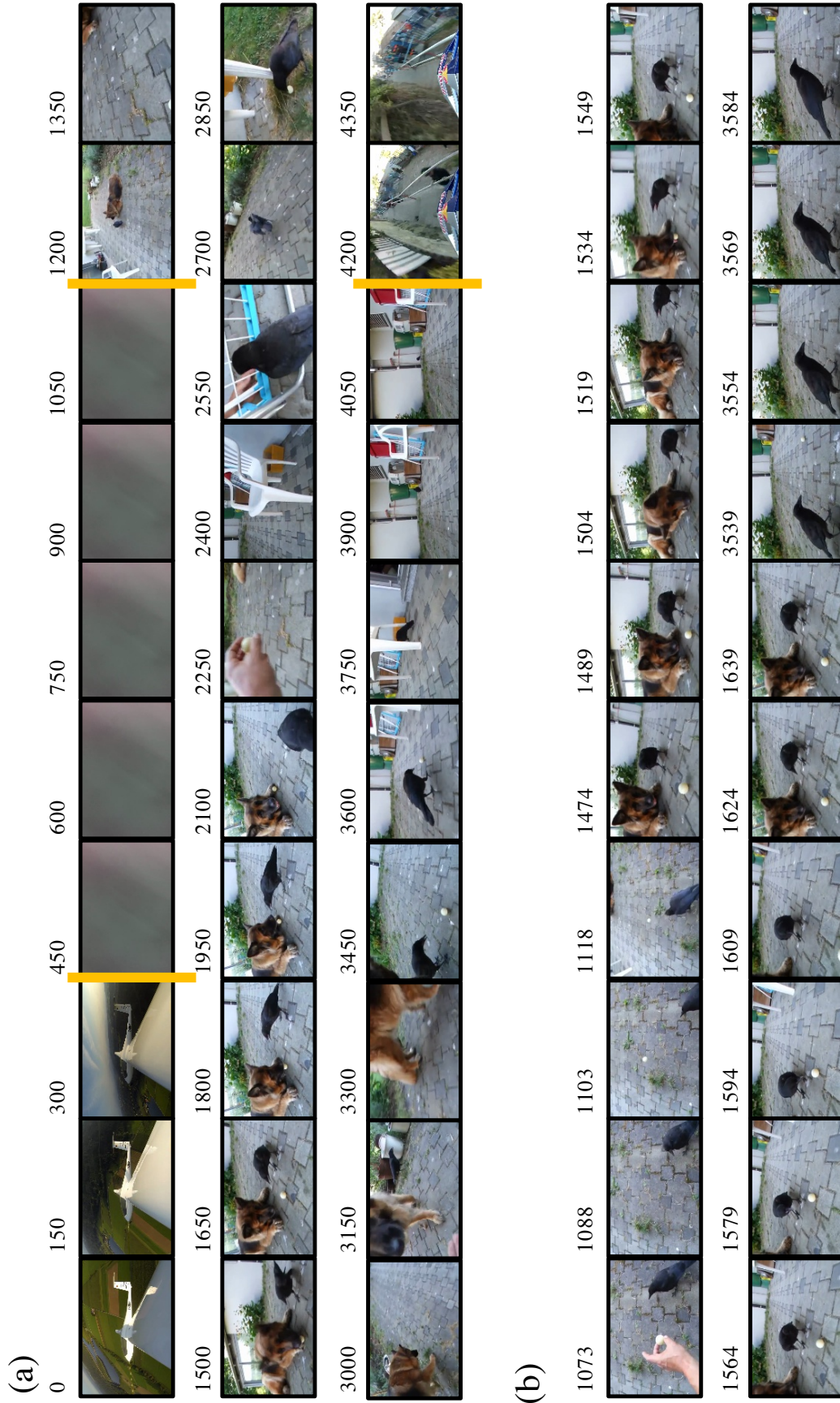


Figure VIII: Machine generated summary of video25 in the IU dataset. (a) The entire sequence of the video. (b) Generated summary by our trained model. Our summary only focuses on the important clip of the video, as expected.