

Supplementary material for AudViSum

Sanjoy Chowdhury*¹

schowdhury671@gmail.com

Aditya P. Patra*²

aditya.prakash.patra1997@gmail.com

Subhrajyoti Dasgupta³

subhrajyotidg@gmail.com

Ujjwal Bhattacharya³

ujjwal@isical.ac.in

¹ ShareChat,

Bangalore, India

² Indian Institute of Technology,

Patna, India

³ Indian Statistical Institute,

Kolkata, India

*indicates equal contributions

1 Audio-Visual Preprocessing

One very crucial aspect of our recent exploration on audio-visual summary generation is the preprocessing step. Unlike previous attempts we consider both audio and visual streams to produce semantically meaningful yet diverse summaries for a given input video. Typical frame level annotation score is deemed unsuitable for this. To ensure proper audio-visual coherence we divide the sequence of data into segments of 3 seconds. This enables us to capture sufficient audio information. After this step, an input video can be converted into a sequence of shot level feature representations. For efficient feature extraction from the audio sequence we need to divide them into meaningful segments and remove the unnecessary intervals. For this, we define the following heuristic, also described in flowchart (Fig. 1).

```
def preprocess(old_list):
    conc_list=[]
    last=[0,0]
    for i in old_list:
        itr=[i[0],i[1]]
        if last==[0,0]:
            if itr[1]-itr[0]>=3*sr: # too big
                conc_list.append(itr)
            else:
                last=itr
        else:
            if itr[0]-last[1]>1.5*sr: # if gap between itr is big not a breathgap
                conc_list.append(last)
                last=itr
            elif itr[1]-last[0]<=3*sr: # if its a breathing gap & combined <= 3sr
                last=[last[0],itr[1]]
            elif itr[1]-last[0]>3*sr:
                conc_list.append(last)
                last=itr
            else:
                print(error)
    if last != [0,0]:
        conc_list.append(last)
        last=[0,0]

    return conc_list
```

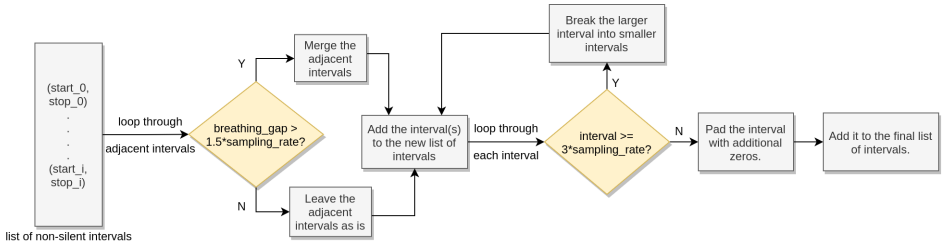


Figure 1: A flow diagram of the preprocessing algorithm applied to the audio segments to eliminate large breathing gaps and split them into consistent and equal lengths.

2 Dataset Annotation

We discuss the TVSum [9] and OVP dataset annotation statistics in this section. The existing annotations were based on visual stimuli only thereby making themselves inappropriate for a study involving a holistic audio-visual summary generation. TVSum was the first to curate a collection of 50 videos and their shot-level importance scores obtained via crowdsourcing. OVP contains videos from various genres with related and meaningful audio content, hence this was considered to be a good benchmark dataset to study the adaptability of an audio-visual summarization model. We chose to ignore another popular dataset SumMe [10] as it does not contain meaningful audio information in the videos. As described in the paper we follow a three pass annotation strategy to make sure that the scoring is done considering both audio as well as visual information.

Fig. 2 presents the distribution of aggregated shot-level scores for each of the three cases separately i.e. when audio-only, visual-only and audio-visual scores are provided. We notice that the majority of shots are provided a score of 1 (least important) whereas, only a small fraction is chosen as important shots (score 5) which are to be considered while generating the summary. Fig. 3 demonstrates distribution of the aggregated shot-level scores for both TVSum and OVP dataset.

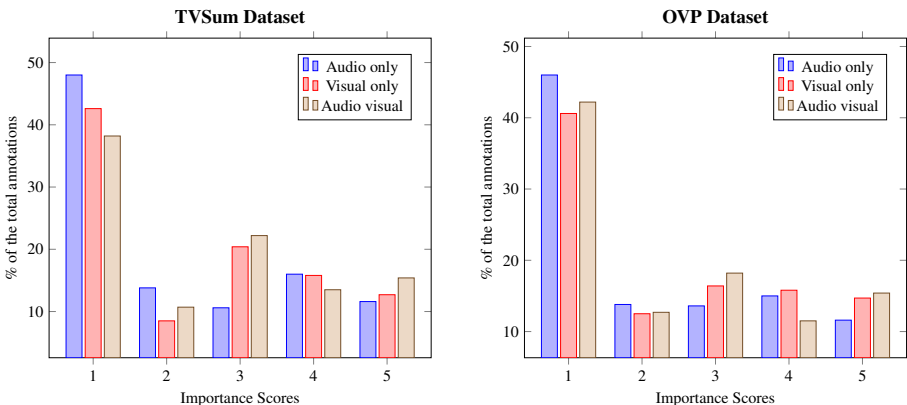


Figure 2: Distribution of the shot-level Importance scores for TVSum and OVP datasets respectively based on corresponding modalities.

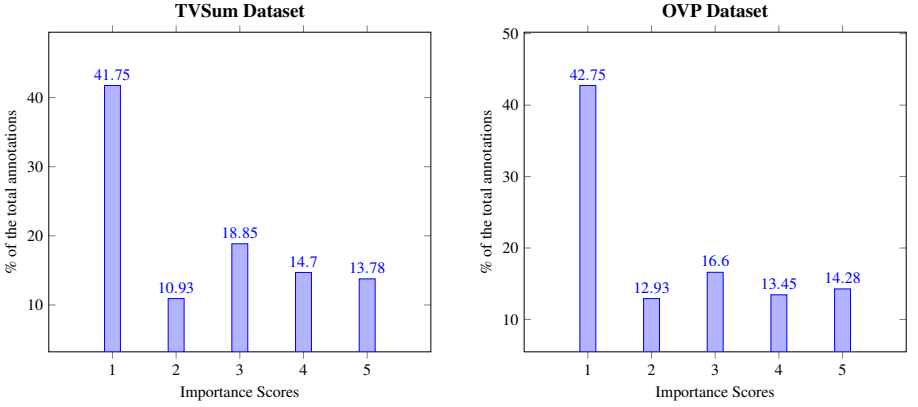


Figure 3: Distribution of the weighted shot-level Importance scores for TVSum and OVP datasets respectively.

3 Training Details

3.1 Training with Policy Gradient

We follow the REINFORCE algorithm [8] to compute the derivative of the objective function $J(\theta)$ w.r.t. the parameters θ as:

$$\nabla_{\theta} J(\theta) = E_{p_{\theta}(a_{1:T})} \left[\mathcal{R}(\mathcal{S}) \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t | h_t) \right] \quad (1)$$

where a_t is the action taken by *AudViSum* at time t and h_t represents the hidden state in BiLSTM.

For the m^{th} model we take the average gradient by running the agent for N_m episodes on the same video. It is computationally efficient as it bypasses the calculations involved to deal with expectation over high dimensional action sequences.

$$\nabla_{\theta_m} J(\theta_m) \approx \frac{1}{N_m} \sum_{e=1}^{N_m} \sum_{t=1}^T \mathcal{R}_e \nabla_{\theta_m} \log \pi_{\theta_m}(a_t | h_t) \quad (2)$$

\mathcal{R}_e is the reward for the e^{th} episode. We can refer to Eq. 2 as the episodic REINFORCE algorithm. The gradient in Eq. 2 might contain high variance resulting in difficulties in model convergence. To alleviate this problem, we subtract the reward by a constant baseline b . Here, b is computed as the moving average of rewards. The revised gradient becomes:

$$\nabla_{\theta_m} J(\theta_m) \approx \frac{1}{N_m} \sum_{e=1}^{N_m} \sum_{t=1}^T (\mathcal{R}_e - b) \nabla_{\theta_m} \log \pi_{\theta_m}(a_t | h_t) \quad (3)$$

3.2 Implementation Details

For training the models we use a Learning Rate (LR) of 0.001, Momentum 0.9 with *Stochastic Gradient Descent (SGD)* optimizer. The proposed method is implemented on a PC with

Intel Xeon E5-2620v3 with 128 GB RAM and a NVIDIA GTX 1080Ti GPU with 12 GB memory. Training is run till 50 epochs unless reward converges. In which case we execute early stopping. The dimension of hidden states in the BiLSTM is set to 384 for audio and 200 for image sequences.

3.3 Regularizer

As selecting more shots increases the overall reward, we apply a regularizer on the probability distributions $p_{1:T}$ to constrain the percentage of shots selected for the summary. We minimize the following term as proposed by [2] during training:

$$L_{\text{percentage}} = \left\| \frac{1}{T} \sum_{t=1}^T p_t - \varepsilon \right\|^2 \quad (4)$$

where ε controls the fraction of shots to be selected. Additionally, we use ℓ_2 regularization on the weight parameters θ to avoid overfitting:

$$L_{\text{weight}} = \sum_{i,j} \theta_{i,j}^2 \quad (5)$$

4 Results

We produce n summaries initially from which top-3 are chosen for final evaluation. It is experimentally observed that the best summary is most likely to be contained within top-3 summaries. Fig. 4 demonstrates the top-3 summaries produced by our self-supervised model for two of the videos. The generated summaries are diverse when compared against each other. Duration of the produced summaries are roughly 15% to that of the input videos.

As the corresponding audio summaries can not be presented in the paper, we share some sample summaries [here](#). We will also release the code used to carry out the experiments to further facilitate reproducibility of our work.



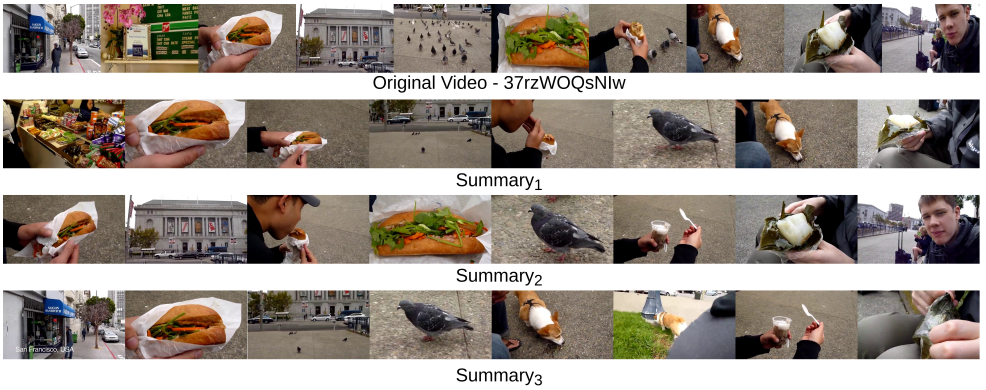


Figure 4: Example videos and their top-3 summaries produced by *AudViSum_{contrastive}*.

5 Evaluation Metric

Otani *et al.* [9] have experimentally shown that Kendall’s τ and Spearman’s ρ correlation coefficients are more robust compared to a classical F1-score when it comes to measuring the strength of correlation between predicted and human annotated video summaries. Since our exploration follows a very similar trajectory we chose to incorporate their findings here. Following their study we use these two metrics for performance evaluation in our paper.

References

- [1] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In *European Conference on Computer Vision (ECCV)*, pages 505–520. Springer, 2014.
- [2] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. Unsupervised video summarization with adversarial lstm networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 202–211, 2017.
- [3] Mayu Otani, Yuta Nakashima, Esa Rahtu, and Janne Heikkila. Rethinking the evaluation of video summaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7596–7604, 2019.
- [4] 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.
- [5] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.