DeepVideo: Video Summarization using Temporal Sequence Modelling

Juhi Naik
Computer Science
Stanford University
juhinaik@stanford.edu

Abstract

In this project we present a tool to generate a summary of the most salient parts of videos. Unlike most research going on in the field of video compression, instead of decreasing redundancy, we try to shorten the video by skipping the "uninteresting" parts. A new approach has been tried for scoring importance of frames. We try 2 models, Convolutional Neural Nets (CNNs) and CNNs combined with Long Short-Term Memory (LSTM) modules and find that the latter works much better on video data. A different cost function was also tried, using Kullback-Leibler divergence to solve the regression problem instead of MSE.

1. Introduction

Given the development in video capturing devices and growing popularity of social media, there are huge volumes of videos being captured and uploaded every second. For example, YouTube has 400 hours of video uploaded every min. One of the most daunting tasks that users face on such sites is to find the interesting/relevant videos from the search results without opening and going through each one. If a short summary of the video could be included with the search results, online surfing could become a much more hassle free and enjoyable experience.

Creating highlights of sports matches or synopses of episodes in TV series are other fields where video summarization plays an important role. However, given the huge volume of online video data, it is infeasible to generate summaries of everything manually. So, in this work, we try to implement a regression model using deep learning methods that attempts to do this automatically.

The input to our algorithm is a video which we then feed to the model as a sequence of frames. A CNN+bidirectional LSTM model is used to output predicted importance scores for each frame. The frames that cross a certain threshold are then stitched together to output the video summary.

2. Related Work

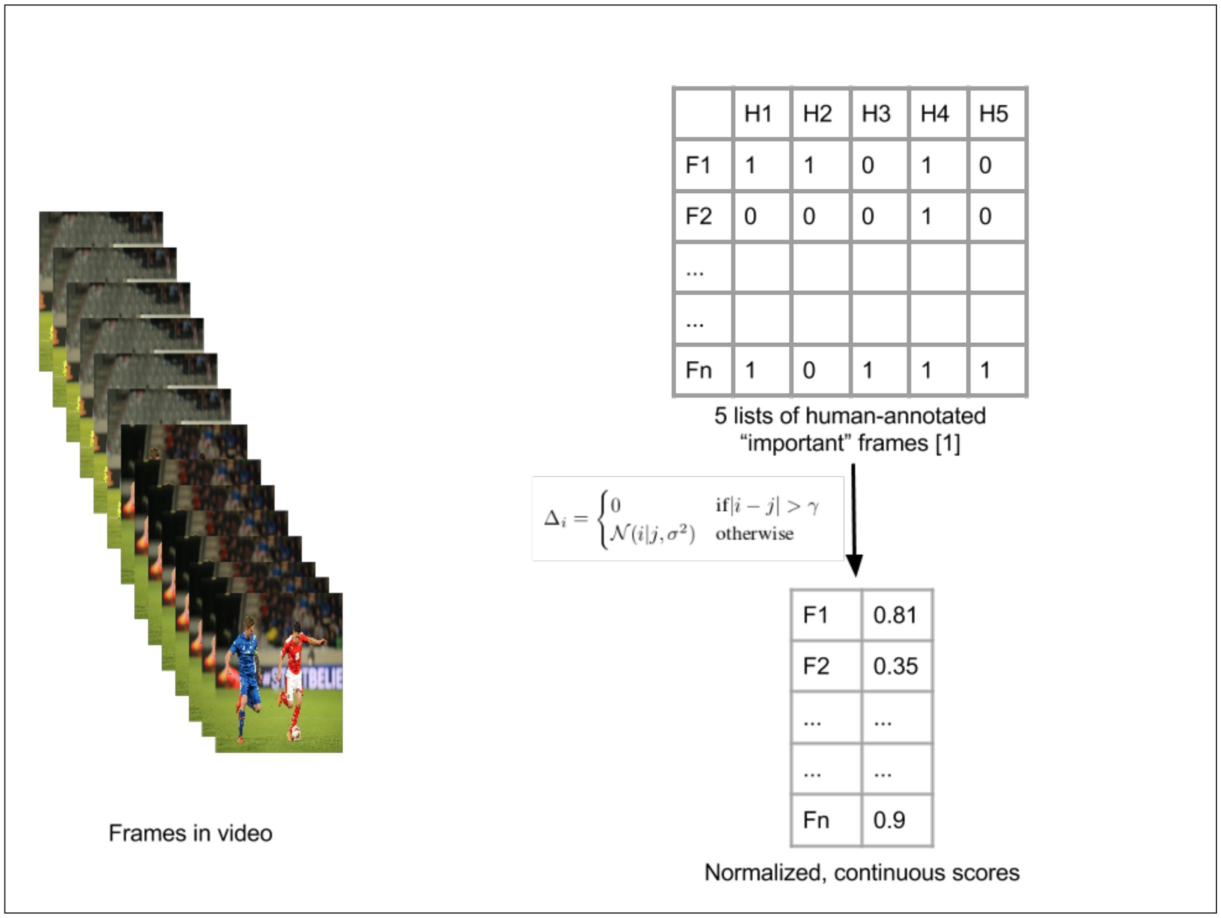
Video Shortening has been a field of active research since a long time. However, the focus has mainly been on either decreasing storage space using compression or removing redundant frames without loss of actual content. The latter is based on extracting key-frames from the video that can best represent a sequence of frames. One of the common approaches to do this is based on frame content changes computed by features, such as color histogram [14] or motion activity [13]. Another very common technique is to cluster the frames using supervised or unsupervised learning by the similarity of their content. Zhuang proposed an unsupervised clustering scheme to adaptively extract key-frames from shots [15]. Other more sophisticated methods include the integration of the motion and spatial activity analysis with face detection technologies [3], a progressive multi-resolution key- frame extraction techniques [1], and object-based approach [6]. The trajectories of objects are used in [11] while user attention is modelled in [12]. The linear dynamical system theory is applied in [9]. Singular value decomposition is adopted to summarize video content in [4].

Advanced Computer Vision Techniques and Deep learning have only recently found their way into this field. [8] combines deep CNNs and RBMs to extract keyframes from videos. [5] uses web images as a prior to rank frames by their significance.

All these techniques concentrate on reducing redundancy in the video while keeping all the content. Another approach possible for video summarization, the one taken in this work, is to identify the "highlights" or the most important frames of the video and only keep them. [7] thresholds frames based on an importance score associated to each. However, the summarization is done on segments of video instead of entire videos, with the segments identified using clustering.

2.1. Dataset

We used the dataset provided by [2]. There are a total of 50 videos in the dataset collected from websites



Frames in video
5 lists of human-annotated "important" frames [1]
Normalized, continuous scores

	H1	H2	H3	H4	H5
F1	1	1	0	1	0
F2	0	0	0	1	0
...					
...					
Fn	1	0	1	1	1

$$\Delta_i = \begin{cases} 0 & \text{if} |i-j| > \gamma \\ \mathcal{N}(i|j,\sigma^2) & \text{otherwise} \end{cases}$$

F1	0.81
F2	0.35
...	...
...	...
Fn	0.9

Figure 1. Each frame in the video is given importance according to how far it is from the annotated frames using a Gaussian distribution

like YouTube. These videos are distributed among several genres (cartoons, news, sports, commercials, tv-shows and home videos) and their duration varies from 1 to 10 minutes. It also contains 250 user summaries, each of which is a list of frames they think are most important in the video. These summaries were created manually by 50 users, each one dealing with 5 videos, meaning that each video has 5 video summaries created by 5 different users. We split the data into 40 videos in the training and 10 in the test set.

To get the importance scores, we apply a Gaussian distribution over each frame that the user annotated as "important" with the frame index as the mean so that the neighboring frames also get some importance to maintain an element of continuity. The final importance scores are normalized so that each value lies between 0 and 1. Given each frame j that was marked important by a user, we calculate the score Δ_i of frame i as

$$\Delta_i = \begin{cases} 0 & \text{if} |i-j| > \gamma \\ \mathcal{N}(i|j,\sigma^2) & \text{otherwise} \end{cases}$$

where, $\mathcal{N}(i|j,\sigma^2)$ represents the normal distribution centered around j with a standard deviation of σ and γ is a window size beyond which we clip the scores to 0.

Using this method, we thus convert the scores to predicted to a continuous distribution between 0 and 1 which is an ideal setting for any regression problem. Also, defining a Gaussian over neighboring frames makes sense because if a particular frame is important, it is probable that the frames just before and after it are also important. This helps ensure that the summary has contiguous sequences of frames instead of disjoint ones.

3. Technical Approach

Convolutional Neural Networks are widely used in a variety of vision tasks. Hence it seemed natural to us to incorporate them in our framework. But the video domain is much more complex than usual images on which convolutional networks have found recent success. In light of this fact, we think that treating this problem as more of a tem-

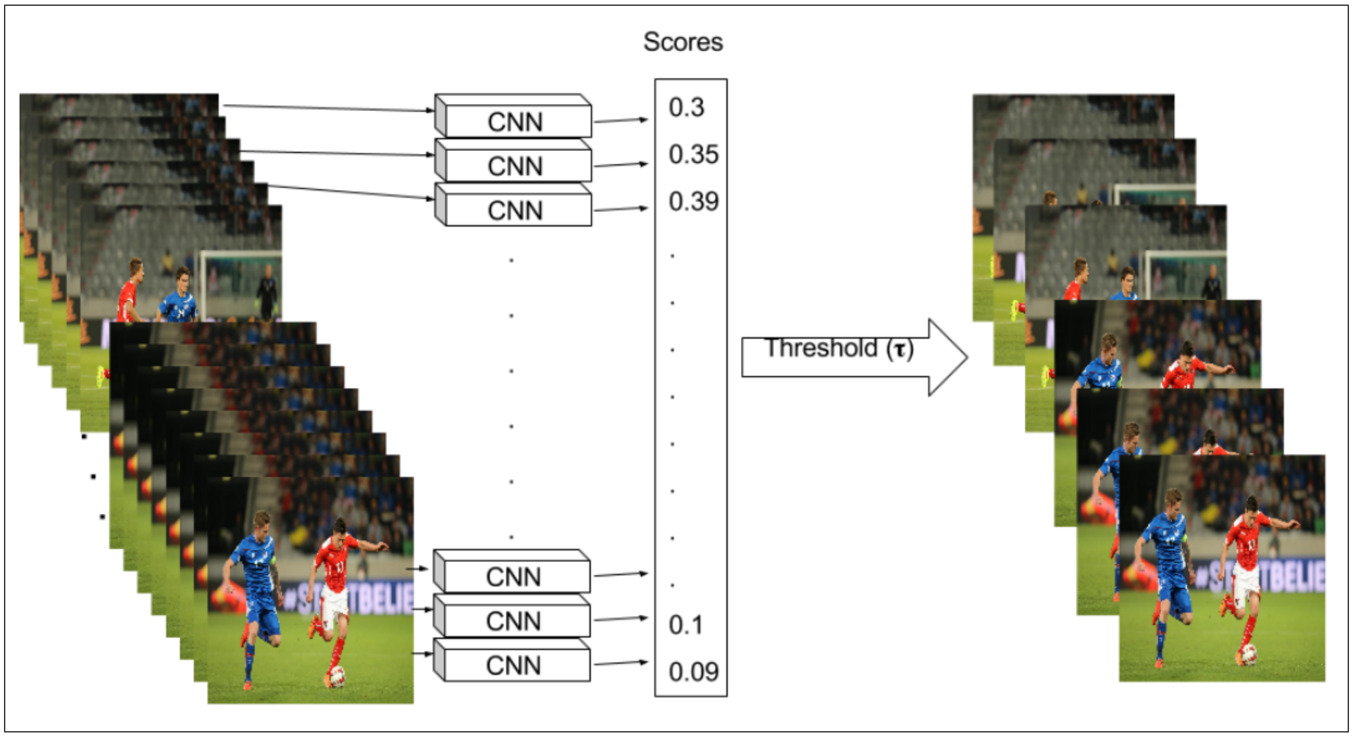


Figure 2. We take a full video as a sequence of frames and run each frame through the trained CNN model. The output received is a vector of size equal to the number of frames in the video, where a single score denotes how important the content in the corresponding frame is. Finally, only the frames having a score above a certain threshold are included in the final summary of the video.

poral sequence modeling problem makes sense. Long Short Term Memories (LSTMs) are the best suited for learning temporal dependencies in the data. Hence a deep learning model that combined the advantages of both seems to be most apt to solve the video summarization task at hand.

3.1. CNN

The most successful model we trained consisting of only CNN was a 7 layered deep network with the following specifications (depicted in Figure 2):-

- 3 x Conv-SpatialBN-ReLU-Pool:
  - Convolutional Layer: 32 Filters of size (3,3) and Stride 1 padded to preserve size of input.
  - Spatial Batch Normalization layer
  - ReLU Activation
  - Max-Pool Layer: (2,2) Filters and stride 2
- 3 x feedforward layers
  - 1st layer: ReLU Activation and 4096 hidden units
  - 2nd layer: ReLU Activation and 1024 hidden units
  - 3rd layer: ReLU Activation and 512 hidden units
- Affine Layer: Sigmoidal Activation and 1 output unit

The final output acts as the ”importance” score predicted for each frame fed into the model.

The model was trained to optimize MSE loss as described in the next section. During test time, the scores generated for the set of frames in a video determine whether the corresponding frames will be included in the final summary or not. Only the frames with scores above a certain threshold are included in the summary. The threshold value can be varied according to the percentage of summarization or duration of summary required.

3.2. CNN + LSTM

Dealing with video data almost always entails some kind of temporal modelling for better results. We decided to do this by adding a bidirectional LSTM layer to our network, in order to capture both the forward and backward dependencies between neighboring frames. Thus, each frame's importance score depended not only on it's own content but also the importance of the frames near it.

Specifically the models used (depicted in figure 3) were as follows:

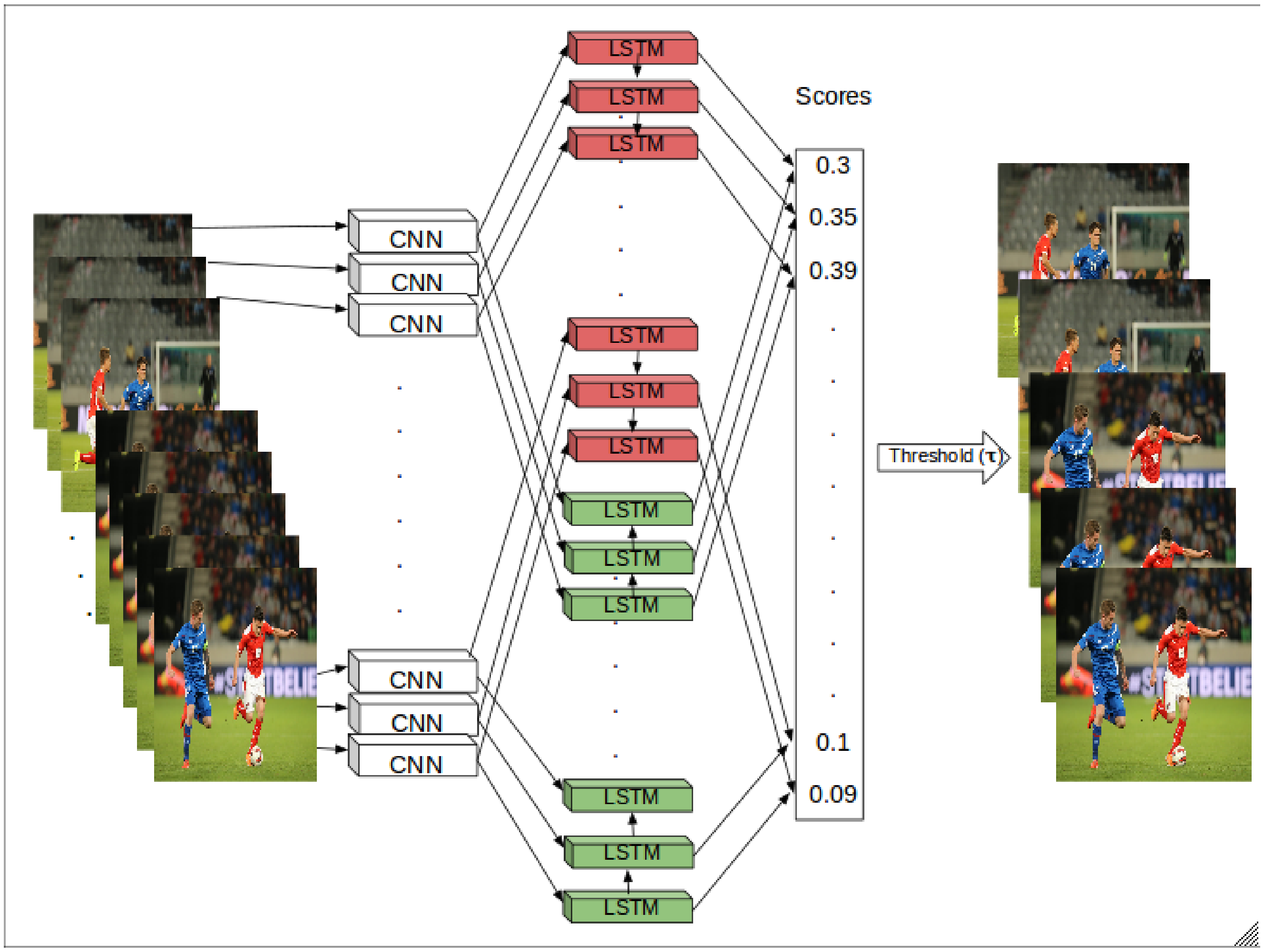


Figure 3. A full video, as a sequence of frames, is passed through the trained CNN + LSTM model. The bidirectional LSTM captures the forward and backward dependencies. The output received is a vector of scores similar to the previous model.

Model 1:

- 3 x Conv-SpatialBN-ReLU-Pool:
  - Convolutional Layer: 32 Filters of size (3,3) and Stride 1 padded to preserve size of input.
  - Spatial Batch Normalization layer
  - ReLU Activation
  - Max-Pool Layer: (2,2) Filters and stride 2
- 2 x feedforward layers
  - 1st layer: ReLU Activation and 256 hidden units
  - 2nd layer: ReLU Activation and 128 hidden units
- 2 x LSTM layers
  - Forward LSTM: 256 units
  - Backward LSTM: 256 units
- Affine Layer: Sigmoidal Activation and 1 output unit

Model 2:

- 3 x Conv-SpatialBN-ReLU-Pool:
  - Convolutional Layer: 32 Filters of size (3,3) and Stride 1 padded to preserve size of input.
  - Spatial Batch Normalization layer
  - ReLU Activation
  - Max-Pool Layer: (2,2) Filters and stride 2
- 2 x feedforward layers
  - 1st layer: ReLU Activation and 256 hidden units
  - 2nd layer: ReLU Activation and 128 hidden units

- 2 x LSTM layers
  - Forward LSTM: 256 units
  - Backward LSTM: 256 units
- Affine Layer: Sigmoidal Activation and 64 hidden units
- Affine Layers: Sigmoidal Activation and k output units

While training, the first model was used to optimize MSE loss while the second was used to optimize the KL-divergence. The number of output units, k, in the final part is determined by the number of parameters required to express the probability distribution of the true scores. Both these loss functions as well as the calculation of probability distribution are described in the next section.

During testing, when the first model was used, the scores were similarly calculated for each frame and the ones above a threshold were included.

When the second model was used, the frames are fed into the model. The scores of each frame are drawn from the probability distribution parametrized by the values that are obtained as output from the model. The final summary, again, contained the frames that received a score above a certain threshold.

4. Loss Functions

4.1. MSE

The Mean-Squared Error (MSE) loss $\mathcal{L}$ was computed as follows:-

$$\mathcal{L} = \frac{1}{N}\Sigma_{i=1}^{N}(\Delta_i - \hat{\Delta}_i)^2 + \lambda||W||_2^2$$

where N is the mini batch size, Δ_i is the true score (ground truth), $\hat{\Delta}_i$ is the score predicted by the model for the i^{th} training example and the second term represents the L2-regularization over all weights.

4.2. Kullback-Leibler Divergence

When we try to minimise MSE loss in this problem, the model tries to match the scores of the frames exactly to the expected ones. However, what is most important to us is that the distribution of scores in the output of the model is similar to the distribution in the expected scores, regardless of what the scores are, specifically.

Thus, to find the difference between the 2 probability distributions, we first fit a Gaussian distribution over the expected scores. The loss function we then use is the KL divergence, $\mathcal{K}_1$ between the 2 normal distributions - the expected one and the one given by the model,

$$\mathcal{K}_1 = \frac{1}{V}\Sigma_{i=1}^{V}\left(\log\frac{\hat{\sigma}_i}{\sigma_i} + \frac{\sigma_i^2 + (\mu_i - \hat{\mu}_i)^2}{2\hat{\sigma}_i^2} - \frac{1}{2}\right)$$

where μ_i, σ_i are the mean and standard deviation of the expected scores of the i^{th} video, $\hat{\mu}_i, \hat{\sigma}_i$ are the corresponding parameters given by the model and V is the number of total videos. If we use this loss function, the k in the second model above would be 2, the parameters being just the mean and the standard deviation.

Given our prior knowledge of the distribution of scores, a Gaussian Mixture Model, which is a mixture of 5 gaussians should fit better on the data. Calculation of the KL divergence between 2 GMMs, however, is intractable in nature. So, the approximation suggested in [10] is used instead to calculate the Symmetric GMM Distance $\mathcal{K}_2$,

$$\mathcal{K}_2 = -\frac{1}{V}\Sigma_{k=1}^{V}\log\left(\frac{2\Sigma_{i,j}\pi_i\pi_j'^k\rho_1}{\Sigma_{i,j}\pi_i\pi_j'^k\rho_2 + \Sigma_{i,j}\pi_i\pi_j'^k\rho_3}\right)$$

where,

$$\rho_1 = \sqrt{\frac{V_{ij}^k}{exp(l_{ij}^k)\sigma_i\sigma_j'^k}};\ \rho_2 = \sqrt{\frac{V_{ij}^k}{exp(l_{ij}^k)\sigma_i\sigma_j^k}}$$

$$\rho_3 = \sqrt{\frac{V_{ij}^k}{exp(l_{ij}^k)\sigma_i'\sigma_j'^k}}$$

$$V_{ij}^k = \frac{1}{\frac{1}{\sigma_i} + \frac{1}{\sigma_j'^k}};\ l_{ij}^k = \frac{\mu_i^k(\mu_i - \mu_j'^k)}{\sigma_i} + \frac{\mu_j'^k(\mu_j'^k - \mu_i)}{\sigma_j'^k}$$

π, π' are the weights, μ, μ' are the means and σ, σ' are the standard deviations of the expected scores the model output respectively.

5. Results and Discussion

The summarization threshold was set so that the summarized video always contained 15% of the total frames in the original video.

Using the CNN model above with the MSE loss function, we get a loss of, $\mathcal{L} = 0.0975$. The learning trend obtained using this method was fairly good. Also, the frames chosen were "correct" when manually inspected. However, due to lack of any temporal information, the frames lacked continuity. So the final summarized video looked more like discrete images stitched together.

On changing the model to include LSTM layers, the MSE loss dropped to 0.083. More importantly, the summarized video looked much more continuous and uninterrupted.

As discussed earlier, optimizing the MSE loss function is not appropriate for the problem statement at hand. It is the distribution of the scores that we want to capture rather than

Model	Loss function	LL
CNN	MSE	0.0975
CNN w/ Bidirectional LSTM	MSE	0.083
CNN w/ Bidirectional LSTM	KL divergence with a Gaussian	1.072
CNN w/ Bidirectional LSTM	Symmetric GMM Distance	1.013

Table 1. Table showing results of various models and loss functions

the exact scores. MSE tends to penalize the output scores unnecessarily even when the underlying distributions match closely. For example, if the location of the peaks correspond in the expected and predicted outputs but the exact height of the peaks don't, the MSE loss would be high.

Using the first KL divergence method described above, the loss obtained was $\mathcal{K}_1 = 1.072$. Also, the quality of the summarization, as per human evaluation, improved.

The best performance was obtained on using a model with CNN and bidirectional LSTMs with a loss function using the symmetric GMM Distance as described above.

The experiments and their results are summarized in Table 1. Examples of original and summarized videos can be found here.

6. Future Work

There are a lot of features that can potentially be added to improve the performance of this model. One of the most significant ones is augmenting with audio data. For example, background noise from the audience in a sports match is a strong indicator of importance of frames. Another feature could be incorporating the rate of re-watching of certain parts of online videos by users which could signify interesting parts of the video.

While most of the evaluation in this work has been human, a potential improvement would be to come up with some kind of metric to measure the performance of the model or the quality of summarization automatically. Also, we can try to come up with better loss functions that are more indicative of our objective.

References

[1] P. Campisi, A. Longari, and A. Neri. Automatic key frame selection using a wavelet-based approach. In *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, pages 861–872. International Society for Optics and Photonics, 1999.

[2] S. E. F. de Avila, A. P. B. Lopes, A. da Luz Jr., and A. de Albuquerque Arajo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56 – 68, 2011. Image Processing, Computer Vision and Pattern Recognition in Latin America.

[3] F. Dirfaux. Key frame selection to represent a video. In *Image Processing, 2000. Proceedings. 2000 International Conference on*, volume 2, pages 275–278. IEEE, 2000.

[4] Y. Gong and X. Liu. Video summarization using singular value decomposition. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 174–180. IEEE, 2000.

[5] A. Khosla, R. Hamid, C.-J. Lin, and N. Sundaresan. Large-scale video summarization using web-image priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2698–2705, 2013.

[6] C. Kim and J.-N. Hwang. An integrated scheme for object-based video abstraction. In *Proceedings of the eighth ACM international conference on Multimedia*, pages 303–311. ACM, 2000.

[7] Y.-F. Ma, L. Lu, H.-J. Zhang, and M. Li. A user attention model for video summarization. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 533–542. ACM, 2002.

[8] O. Morere, H. Goh, A. Veillard, V. Chandrasekhar, and J. Lin. Co-regularized deep representations for video summarization. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 3165–3169. IEEE, 2015.

[9] X. Orriols and X. Binefa. An em algorithm for video summarization, generative model approach. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 335–342. IEEE, 2001.

[10] G. Sfikas, C. Constantinopoulos, A. Likas, and N. P. Galatsanos. An analytic distance metric for gaussian mixture models with application in image retrieval. In *Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005*, pages 835–840. Springer, 2005.

[11] A. Stefanidis, P. Partsinevelos, P. Agouris, and P. Doucette. Summarizing video datasets in the spatiotemporal domain. In *Database and Expert Systems Applications, 2000. Proceedings. 11th International Workshop on*, pages 906–912. IEEE, 2000.

[12] S. Uchihachi, J. T. Foote, and L. Wilcox. Automatic video summarization using a measure of shot importance and a frame-packing method, Mar. 18 2003. US Patent 6,535,639.

[13] W. Wolf. Key frame selection by motion analysis. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 2, pages 1228–1231. IEEE, 1996.

[14] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern recognition*, 30(4):643–658, 1997.

[15] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, volume 1, pages 866–870. IEEE, 1998.