SumBot: Summarize Videos Like a Human

Hongxiang Gu
University of California, Los Angeles
hxgu@cs.ucla.edu

Stefano Petrangeli
Adobe Research
petrange@adobe.com

Viswanathan Swaminathan
Adobe Research
vishy@adobe.com

***Abstract*—Video currently accounts for 70% of all internet traffic and this number is expected to continue to grow. Each minute, more than 500 hours worth of videos are uploaded on YouTube. Generating engaging short videos out of the raw captured content is often a time-consuming and cumbersome activity for content creators. Existing ML-based video summarization and highlight generation approaches often neglect the fact that many summarization tasks require specific domain knowledge of the video content, and that human editors often follow a semi-structured template when creating the summary (e.g. to create the highlights for a sport event). We therefore address in this paper the challenge of creating domain-specific summaries, by actively leveraging this editorial template. Particularly, we present an Inverse Reinforcement Learning (IRL)-based framework that can automatically learn the hidden structure or template followed by a human expert when generating a video summary for a specific domain. Particularly, we propose to formulate the video summarization task as a Markov Decision Process, where each state is a combination of the features of the video shots added to the summary, and the possible actions are to include/remove a shot from the summary or leave it as is. Using a set of domain-specific human-generated video highlights as examples, we employ a Maximum Entropy IRL algorithm to learn the implicit reward function governing the summary generation process. The learned reward function is then used to train an RL-agent that can produce video summaries for a specific domain, closely resembling what a human expert would create. Learning from expert demonstrations allows our approach to be applicable to any domain or editorial styles. To demonstrate the superior performance of our approach, we employ it to the task of soccer games highlight generation and show that it outperforms other state-of-the-art methods, both quantitatively and qualitatively.**

I. Introduction

Video watching is one of the most popular digital activities worldwide. A large amount of video content is captured, produced and distributed over the Internet every minute and this number has been growing rapidly over the past few years. Creating effective and engaging videos that can resonate with the target audience is fundamental for the success of content creators. Particularly, generating short summaries or highlights of the captured video content is an essential task not only for content publishing on video sharing platforms, but also for video asset management and video advertising purposes. The popularity of short video sharing platforms (like Tik Tok and Instagram) is constantly accelerating the life cycle of a video, therefore creating a huge demand for fast and high-quality automated video summarization algorithms. For this reason, in recent years, several ML-based algorithms have been proposed to support content creators and automate this task.

ML-based video summarization algorithms aim to produce a short and insightful summary starting from a long video that often contains superfluous content that is likely to be not interesting for the viewer. For example, given an NBA basketball game, we might be interested in automatically generating a 5-10 minutes summary with the most important moments. Several algorithms have been proposed in the past to solve this problem [1], [2], [3], but the state-of-the-art methods are still far from achieving human-like performance. Particularly, previous works in this domain often overlook the fact that many human generated summaries follow a semi-structured editorial template (as in the basketball example mentioned above). These templates, utilizing content-specific domain knowledge, enable content creators to achieve consistency and efficiency during the production process. Manually hard-coding these templates for automatic summary generation often requires a unique set of ambiguous rules based on domain knowledge and human intuition.

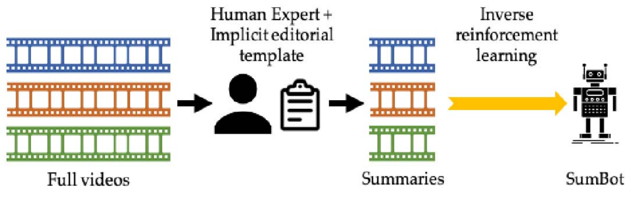

Human Expert + Implicit editorial template
Inverse reinforcement learning
Full videos
Summaries
SumBot

Fig. 1: Given editorial summaries generated by human experts following an implicit, semi-structured editorial template, our SumBot framework employs an Inverse Reinforcement Learning approach to estimate this template, in the form of a reward function, which can then be used to mimic the editing behavior of the human experts.

In this paper, we therefore propose to actively leverage these editorial templates to automatically create high-quality, domain-specific summaries. Particularly, we present the *SumBot* – from *Summarization Bot* – framework (Figure 1). Our proposed framework can automatically learn the domain-specific implicit set of rules (i.e., the template) followed by human experts when creating a summary. For this purpose, we introduce an end-to-end learning framework using an Inverse Reinforcement Learning (IRL) approach that can learn the set of rules, in the form of a reward function, from a small number of human demonstrations. Once this reward function has been learned, it can be used to generate human-like domain-specific summaries. The main contributions of this paper are therefore threefold:

- We design the *SumBot* framework, an end-to-end video summarization framework that is applicable to any summarization task following a (semi-) structured editorial template. Our framework can produce high quality video summaries that mimics the behavior of human editors;
- In the SumBot framework, the video summarization problem is formulated as a Markov Decision Process (MDP) with clear and simple system dynamics, where each state contains the shots of the video included in the summary, and whose actions are to include/remove a shot from the summary, or leave it as is;
- Instead of using handcrafted reward functions, the reward functions in our MDP problem formulation, which encode the editorial template, are learned automatically through human expert demonstrations using an IRL approach. Our technique can therefore learn to summarize any domain or any editorial styles automatically from expert examples.

The remainder of this paper is structured as follows. Section II presents related work in the domain of video summarization algorithms. Section III presents the proposed SumBot framework and, particularly, the modeling of the summarization task as an MDP and the IRL algorithms used to estimate the implicit reward function. Qualitative and quantitative results are presented in Section IV, while Section V concludes the paper and presents several directions for future research.

II. Related Work

Several methods such as Delaunay triangulation [4], K-means clustering [5], STIMO [6], VSCAN [1] use shallow visual features such as color histogram to cluster video frames and draw a sample from each cluster as the summary. In recent years, the development of deep learning has enabled more efficient video summarization methods. Zhang et al. [7] propose to use deep frame-level visual features and a long-short-term-memory-based deep neural network to perform supervised learning on video summarization. Mahasseni et al. use generative adversarial networks to train a deep neural network that summarizes videos in an unsupervised manner [2]. Similarly, Gu et al. propose an unsupervised summarization method based on video reconstruction based on selected video segments [3]. Zhou et al. employ deep reinforcement learning (RL) to summarize a video by maximizing the diversity and representativeness of the selected shots in the summarization process [8]. Jung et al. propose a simple yet effective regularization loss term, called variance loss, to effectively learn the feature importance score of each frame [9]. The authors also design a novel two-stream network named Chunk and Stride Network (CSNet) that utilizes local (chunk) and global (stride) temporal views on the video features to perform the summarization tasks. Yao et al. divide the input video into shots, each analyzed by a different network stream [10]. The spatial stream tries to capture the frame appearance, while the temporal stream the motion dynamics in the clip. Gunawardena et al. use frame level statistics and data clustering to decide what parts of the video should be summarized [11]. A combination of frame-level statistics and hierarchical clustering allow to cluster the different shots, based on color and motion information. Xiong et al. propose an unsupervised method where the video duration is used as a feedback signal on the summary's quality [12]. Given a set of user-generated videos, the main idea is that short videos are more likely to be highlights than long ones. This provides an implicit feedback on which shots should be included in the summary. To solve the lack of annotated human labels for training, Kim et al. use Web images as weak-supervision signals [13]. Jiao et al. introduce a summarization architecture that exploits spatial-dependent video features [14]. An attention mechanism is employed to focus on the regions of the video with high response value because, for example, of an important object in the foreground.

Recently, generating video summaries/highlights using reinforcement learning has attracted the attention of numerous researchers. Lan et al. propose FF-net, which uses reinforcement learning (with handcraft rewards) to fast forward a subset of the video to keep only the essential highlights and form a video summary [15]. Chen et al. has proposed to break the reinforcement learning problem for video summarization into smaller sub-problems, and use hierarchical reinforcement learning to simplify the learning process [16]. Wang et al. perform a deeper investigation by proposing new reward functions to be used in their Deep Summarization Network (DSN), a network that treats the summarization problem as a process of sequential decision making tasks [17].

Even though the problem of video summarization has been studied for many years both in a supervised or unsupervised manner, little effort has been put in learning how an ML-agent can mimic the style of human editors when creating a video summary. More specifically, many studies that formulate video summarization as a sequential decision making problem never intended to learn and understand how human editors perform the task. Instead, most of the existing works that attempt to solve the problem using RL techniques design hand-crafted reward functions and apply these intuition-driven reward functions to an RL-agent. However, the summarization process of human experts is often implicit, and it is hard to determine whether a hand-crafted reward represents this process well or not. To address these problems, we propose the SumBot framework, which learns the structure guiding the summarization process of a human editor, making use of an IRL approach. This choice allows our framework to create video summaries mimicking the behavior of human experts on domain-specific videos.

III. The SumBot Framework

We present in this section the main components of our end-to-end SumBot learning framework. The task of video summarization can be formatted in a variety of different frameworks from supervised learning to unsupervised learning. In this paper instead, we consider the problem as an instance of the learning from demonstration problem. Particularly, we focus on those video summarization tasks where a human editor follows a set of semi-structured rules or editorial templates to create a summary. For example, this is often the case for sports highlight generation. We propose to exploit this unique characteristic and automatically learn the implicit template, using the human-generated highlights as demonstrations for our IRL agent. Compared to other methods that would need to design a handcrafted loss function or obtain a large amount of training data in different domain-specific summarization tasks, our method proves to be more scalable.

The remainder of this section details the different modules of the SumBot framework. First, we present the shot splitting and feature extraction modules. Second, we present the MDP formulation we use to model the video summarization problem and the IRL-based algorithm used to estimate the implicit reward followed by the human experts. We finally indicate how the learned reward can be employed to generate new summaries.

A. Shot Splitting and Feature Extraction

In our context, a shot is defined as a short sequence of consecutive frames that are perceptually consistent, usually a few seconds long. We consider a shot as a good indication of a small event in the video that might be included in the summary. We use a variation of the perceptual hash algorithm called *difference hash* (dHash) [18]. dHash generates a fingerprint of a media file considering multiple visual features, and is guaranteed to generate hashes with small hamming distance if multiple media files are perceptually similar. We calculate the difference hash for all the video frames and perform a split whenever the hamming distance between the hash of two consecutive frames is above a pre-defined threshold.

After obtaining the shot boundaries, we propose to extract shot-level features that can succinctly represent the content of a shot. Particularly, the features for each shot consist of three components: visual features, audio features and classification labels. Frame-level visual features are extracted using a state-of-the-art Inception network [19] trained on ImageNet [20]. The final shot-level visual features are obtained by aggregating all the visual features of the individual frames in the shot. Audio features are extracted using a deep 1D convolutional neural network as described by Aytar et al. [21], by directly applying the raw wave form to the model.

Classification labels are useful in the summarization task because an expert editor will follow specific template rules when editing the video (e.g., for a soccer video to highlight the kick-off, yellow/red cards, goal shots, audience cheering etc). Moreover, the set of possible shot types is usually finite for a given domain. We therefore propose a supervised approach to classify each shot based on the visual and audio features presented above. As we will detail in Section IV, the deep neural network presented in Figure 2 is used to classify the shots of a soccer match in nine categories based on the audio and visual features of the frames in the shot. The classifier we use was originally trained on the Kaggle Football Events dataset [22] with 900,000 events from 9,074 football games.

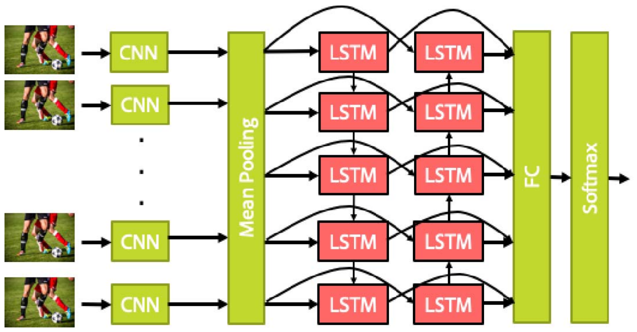


Fig. 2: Video shot type classification neural network architecture.

B. MDP Formulation

We propose to formulate the video summarization task using as a Markov Decision Process. The summarization agent is sequentially presented with all the shots composing the video, from beginning to end, and it needs to decide whether the shot should be included in the summary or not[1]. We consider a video composed of T shots and indicate with F_0^{t-1} the list of features (as described in Section III-A) of all the shots the agent has selected for the summary, from the beginning of the video up until shot t. f^t represents the features of shot t – the shot the agent is currently analyzing. Given the above, the state of the MDP at shot t is defined as

$$s^t = (F_0^{t-1} || f^t) \tag{1}$$

Each state therefore represents the status of the summary up until shot t and the shot the agent is currently considering for the summary. The MDP always starts with a starting state where F_0 is an empty vector and f^0 represents the features of the first shot of the video. The MDP terminates at the termination state where F_0^T is the final summary and f^{T+1} is an empty vector.

At any given state s^t, the agent can take one of the following three actions:

- a_d: Discard the current shot and maintain the summary as is;
- a_s: Select the current shot and add it to the summary;
- $a_{e,s}^i$: Evict the i_{th} shot from the selected summary and add the current shot to the summary.

Based on the selected action, F_0^t is updated accordingly and the agent can move onto the next state. Under the proposed MDP problem formation, the transition from a state $s^t \times a^t \rightarrow s^{t+1}$ is deterministic. We indicate with $F_0^{t-1} = \{\varsigma_0, \ldots \varsigma_k\}$ the summary up until the t_{th} shot, where ς_i is the i_{th} selected shot included in the summary and $k \leq t$. Equation 2 encodes the state transitions in our MDP,

[1]In the case of a live streaming video, the video splitting and feature extraction operations can be performed on the fly as the video stream is transmitted in real time.

$$s^t \times a_d \to s^{t-1} \tag{2a}$$

$$s^t \times a_s \to (\{\varsigma_0, \dots \varsigma_k, \varsigma_t\} || f^{t+1}) \tag{2b}$$

$$s^t \times a_e^i \to (\{\varsigma_0, \dots \varsigma_k, \varsigma_t\} - \{\varsigma_i\} || f^{t+1}), \varsigma \in F_0^t \tag{2c}$$

The reward function is initially unknown. It is worth noting that, in our MDP framework, the reward function indicates the implicit "rationale" guiding the summarization process of the human experts, for a given domain. Estimating this reward would therefore give us the capability to recreate the summarization process of the experts and mimic their behavior. To solve this problem, we propose to use an Inverse Reinforcement Learning approach to learn the reward function using the highlights generated by the human experts as demonstrations.

C. Reward Function Learning using IRL

Estimating the reward function under an IRL setting is known to be an underdefined problem: multiple reward functions can explain the expert demonstrations [23]. In other words, even if we would add more demonstrations, there will always be multiple reward functions that would describe the behavior of the experts. To solve this reward ambiguity problem, we use the Maximum Entropy Inverse Reinforcement Learning (MaxEnt IRL) algorithm proposed by Ziebart et al. [23]. In a nutshell, the MaxEnt IRL algorithm estimates the best reward function that explains the expert demonstrations while maximizing entropy, i.e., making the fewest assumptions about the structure of the data.

We indicate with $\tau = \{s^0, a^0, \dots, s^t, a^t, \dots, s^T, a^T\}$ a trajectory demonstration, and with $D : \{\tau_i\} \sim \pi^*$ the whole set of expert demonstration trajectories, which we assume to be drawn from the optimal policy[2] π^*, and it is unknown at the beginning. We consider the class of reward functions that are linear in the feature space of the states and parameterized by θ, which we want to estimate, as described in the following:

$$R_\theta(\tau) = \sum_t r_\theta(s^t) = \sum_t \theta^\top h^{s^t} \tag{3}$$

where h^{s^t} represents the features of state s^t. While this is a strong assumption, it has been shown to produce good results in practice, as it will be presented in Section IV. Such a formulation also allows us to avoid overfitting, which is particularly important when the number of demonstrations is limited, as in our scenario. Deep neural networks can also be used to approximate complicated and nonlinear reward functions, as detailed by Wulfmeir et al. [24]. In MaxEnt IRL, we assume that the probability of a certain trajectory under the expert is exponential in the reward:

$$P(\tau) = \frac{1}{Z(\theta)} \exp R_\theta(\tau) \tag{4}$$

Intuitively, this means that trajectories with the highest rewards are the most likely to be sampled from the expert. In other words, the higher the reward associated to a trajectory, the higher the chance the trajectory is drawn from the optimal policy. $Z(\theta)$ is a partition function that serves as a normalization factor in the equation. In general, estimating the partition function is hard for continuous or large state spaces with unknown or not fully exposed system dynamics. Since the system dynamics in our MDP formulation is known and deterministic (see Equation 2) and states are discrete, we can estimate the partition function based on sampled expert demonstrations as in the following:

$$Z(\theta) = \sum_{\tau \in D_{sample}} \exp R_\theta(\tau) \tag{5}$$

To estimate the reward function, we maximize the likelihood of our set of expert demonstrations with respect to θ:

$$\max_\theta L(\theta) = \max_\theta \sum_{\tau \in D} \log P_{r_\theta}(\tau) \tag{6}$$

which can be rewritten as:

$$\max_\theta L(\theta) = \sum_{\tau \in D} R_\theta(\tau) - M \log \left(\sum_{\tau \in D} \exp R_\theta(\tau) \right) \tag{7}$$

where M is the number of demonstrations. Finally, this optimization problem can be solved using gradient descent:

$$\nabla_\theta L(\theta) = \sum_{\tau \in D} \sum_{t \in \tau} h^{s^t} - M \sum_{\tau \in D} \sum_{t \in \tau} P(s^t|\theta) h^{s^t} \tag{8}$$

$P(s^t|\theta)$ is the probability of visiting a certain state under the expert demonstrations and a reward function parameterized by θ. This entails that, for every iteration of gradient descent, the optimal policy $\pi(a|s)$ with respect to the current estimation of r_θ has to be computed. We use the fitted value iteration algorithm to obtain this optimal policy [25], from which we can obtain the state visitation probabilities. In a nutshell, the algorithm employed in this paper can be summarized in the following five steps:

1) Gather demonstrations D and initialize θ;
2) Solve for optimal policy $\pi(a|s)$ with respect to the current estimation of r_θ using fitted value iteration;
3) Solve for state visitation frequencies $P(s^t|\theta)$;
4) Compute gradient $\nabla_\theta L$ as in Equation 8;
5) Update θ with $\theta := \theta - \alpha \nabla_\theta L$, with α the learning rate, and repeat from step 2.

The optimal policy and reward function calculated in the last iteration of training are used for inference, as described in Session III-D.

D. Apply the Reward Function

Once the reward function has been learned, we can use it in a standard MDP setting to generate the summary for a video in the same domain as that of the human demonstrations used for training. Particularly, we use the learned reward to generate an optimal policy (e.g., the summary) for any input video; F_0^T

[2]In our setting, we assume expert demonstrations are always optimal.

Algorithm 1: Trajectory construction algorithm.

```
Inputs. The full match video $V_{full}$, and $n$ highlights
  generated by editors on the same match
  $V_{sum} = \{V_0 \dots V_{n-1}\}$, shot comparison threshold $\rho$
  and maximum highlight duration $l$.
Output. An aggregated trajectory $\tau$.
Goal. Generate a demonstration of highlight generated
  on $V_{full}$.
Algorithm:
  1) Preparation
      a) Split shots on $V_{full}$ and each $V_{sum}$ using the
         method introduced in Section III-A;
      b) Generate features for each shot in $V_{full}$ and each
         shot in $V_{sum_0} \dots V_{sum_n}$, denoted as
         $\mathbb{F}_{full} = \{f^0_{full} \cdots f^T_{full}\}$, and
         $\mathbb{F}_{sum_i} = \{f^0_{sum_i} \cdots f^k_{sum_i}\}$, as indicated in
         Section III-A.
  2) Shot matching
      a) Identified the shot overlaps among $V_{sum}$, we
         calculate the importance score $\lambda_i \in \Lambda$ for shot $i$
         as the frequency that shot $i$ appears in all
         highlights $\in V_{sum}$. Shot comparison is performed
         based on frame matching, if the frames shared
         between two shots exceed a threshold $0 < \rho \leq 1$,
         they are considered identical;
      b) Initialize the state $s^0 = (\{\} \| f^0_{full})$;
      c) Traverse each shot in $V_{full}$ in chronological
         order:
          i) if $\lambda_t = 0$, the current step is treated as $s^t \times a_d$
             (i.e., current shot has been discarded by the
             expert as it is not included in the summary);
          ii) if $\lambda_t > 0$ and $\|F^{t-1}_0\| < l$, the current step is
             treated as $s^t \times a_s$;
          iii) if $0 < \lambda_t \leq 1$ and $\lambda_t > \min(\Lambda)$ and
             $||F^(_0 t-1)|| \geq l$, the current step is treated as
             $s^t \times a^i_e$. $f^i$ is evicted, where $\lambda_i = min(\Lambda)$. $f^t$
             is added to the highlight.
  3) Demonstrated trajectory construction
      a) Output $\tau = \{s^0, a^0, \dots, s^t, a^t, \dots, s^T, a^T\}$.
```

containing the features of the summary generated by the agent. We select the corresponding video shots and concatenate them in temporal order to generate the final summary.

E. Preparing Demonstrations for Training

To train our SumBot agent, we need to gather and prepare expert demonstrations for training. When considering existing editorial highlights as demonstration trajectories, we need to take into account that, in most cases, the full trajectory is not directly available. We therefore use a shot matching method by actively comparing F^{t-1}_0 (the features of all the shots included in the summary until time step t), between the editorial summary and the original full video footage. The state matching procedure used in this paper is described in Algorithm 1.

IV. Evaluation

To evaluate the SumBot framework, we apply it to a popular summarization task: generating short summaries of soccer games. Regular soccer games lasts about 90 minutes, and they are usually summarized in short highlights of about 2 to 5 minutes. Soccer games highlights tend to follow a specific template that focuses on the most important moments of the game (as kick-off, goals, yellow/red cards etc.), which makes this summarization application a good candidate to showcase the performance of the proposed approach. We gather original soccer game footage from 120 soccer games from professional YouTube accounts[3] and the corresponding editorial highlights. These high-quality highlights are generated by human experts with a deep understanding of the task at hand, and can therefore be considered optimal. The summary for each game corresponds to one trajectory demonstration used to train our SumBot agent. We construct the demonstration trajectories for training using Algorithm 1 for 100 soccer matches. The remaining 20 highlights are used to test the policy and reward learned by the agent.

In the remainder of this Section, we will present details on metrics and comparison algorithms (Section IV-A), and quantitative and qualitative results (Section IV-B).

A. Metrics and Baseline Methods

To quantify the performance of the proposed method, we propose to use precision and recall for each generated summary. We indicate with $V_{SummBot}$ and V_{Expert} the set of video shots included in the summary by SumBot and the human expert, respectively. In this context, precision P and recall R are calculated as follows:

$$P = \frac{V_{SummBot} \cap V_{Expert}}{V_{SummBot}} \tag{9}$$

$$R = \frac{V_{SummBot} \cap V_{Expert}}{V_{Expert}} \tag{10}$$

Moreover, we also provide the F-score when comparing our approach with other state-of-the-art methods:

$$F_{score} = 2\frac{PR}{P+R} \tag{11}$$

We evaluate the performance of our proposed framework against two state-of-the-art methods proposed by Zhang et al. [7] and by Zhou et al. [26]. Zhang et al. [7] are the first to propose an LSTM network for video summarization, which they combine with a determinantal point process to increase the diversity of selected frames/shots in the final summary. This work is a good example of classical summarization

[3]FIFATV, NBC Sports, Fox Soccer, Champions League on CBS Sports and the official soccer club accounts

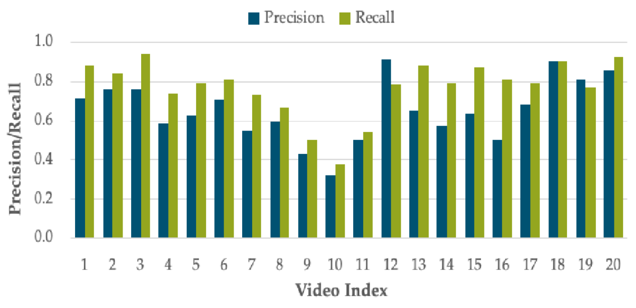


Fig. 3: Evaluation of the proposed SumBot framework on 20 full soccer match videos.

TABLE I: Comparison with other state-of-the-art methods on video summarization for soccer games (average F-score across 20 games).

Method	*F-score*
Zhang et al. [7] – LSTM + Diversity	42.28
Zhou et al. [26] – Deep Q-Learning	58.45
SumBot	**70.65**

methods that consider video-level characteristics to create a summary. The approach proposed by Zhout et al. [26] is closer to that used in this paper. Particularly, the authors set the video summarization task as a sequential decision process, where a reinforcement learning agent has to decide what frames/shots should be included in the summary. A deep Q-Learning approach is used to approximate the action-value function [27]. Similarly as in our framework, the authors employ both frame-level features and video-level semantic category labels. The goal is to include video shots in the final summary that contribute to recognize the final summary in the same category labels as the input video.

B. Results

Quantitative results. Figure 3 reports the results, in terms of precision and recall (see Equations 9 and 10), for the SumBot framework on the summaries generated on 20 full soccer game videos. Higher recall values are due to the fact that we restrict our summary to be 3 minutes long, whereas some editorial summaries are shorter than 3 minutes. For about 15 of the 20 videos, our method is able to reach precision and recall values above 0.6, which indicates a good correspondence with the summary generated by the human experts. It is also worth stressing that this result is achieved by only considering 100 training examples, albeit high-quality.

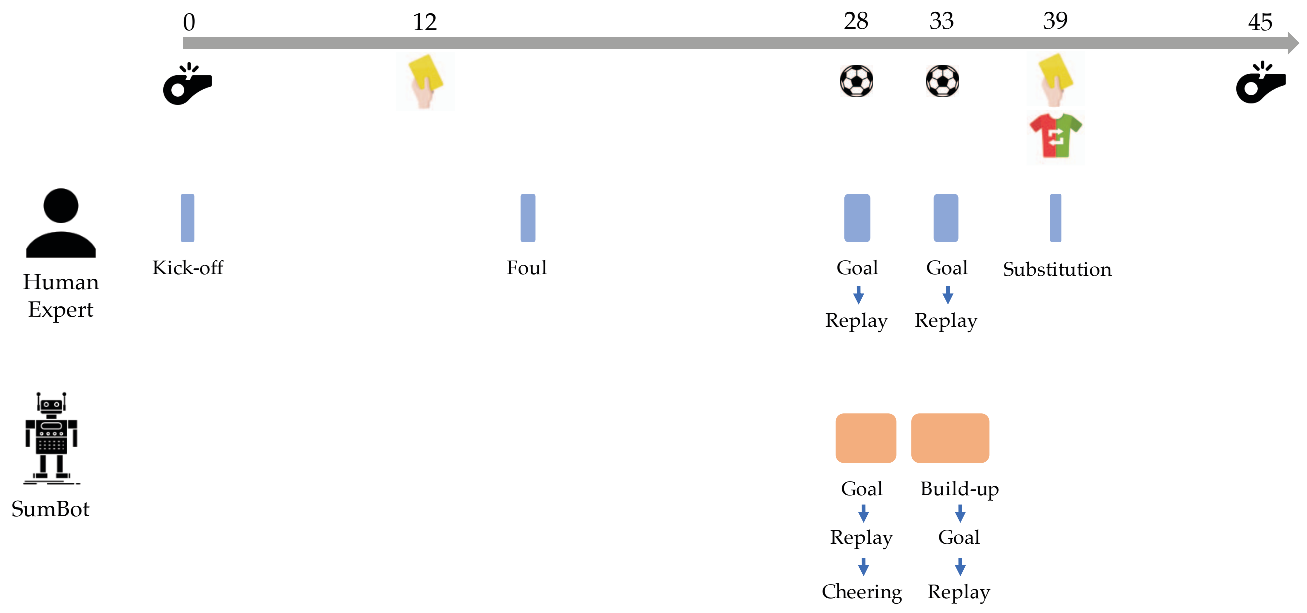
In Table I, we compare our results with the methods proposed by Zhang et al. [7] and Zhou et al. [26], in terms of average F-score (Equations 11) on the 20 soccer game summaries. Our approach is able to outperform the other baseline algorithms by a consistent margin, showing how SumBot can generate summaries in this domain that are much closer to what a human expert would create. The deep Q-Learning-based approach by Zhou et al. [26], which is also setup as a reinforcement learning problem and includes semantic information in the process, reaches better performance than the classical deep-learning method by Zhang et al. [7], which only considers frame-level features to create the summary.

Qualitative results. We present in Figure 4 a qualitative analysis of a summary generated for a soccer game by both a human expert and our SumBot approach. In this case, the human-generated summary is about 2 minutes long, while the summary generated by SumBot is 3 minutes long. Light blue and light orange segments indicate the portion of the original match that has been included in the summary by the expert and Sumbot, respectively. The major events occurring during the match are captured on the gray timeline, both for first (Figure 4(a)) and second half (Figure 4(b)) of the game. We can notice how both summaries include goal shots, which are clearly the most important moments of the match. Similarly as for the human expert, our SumBot agent includes the actual goal shot, together with replay, build-up and cheering shots. In this case study, the human-generated summary introduces more variety, especially for the first half of the match, by including the kick-off shot (minute 0) and a substitution (minute 39). It is worth noting that these moments are included in the summary because they have a particular emotional/sportive value. Our SumBot agent tends to focus more on the goal shots compared to the human expert in this case, which can also be explained by the different duration of the two summaries. Overall, even though the human expert shows more diversity in the highlight process, our SumBot agent is capable of retaining the most important moments of the match and generate a visually-appealing final result that is in line with that generated by the expert.

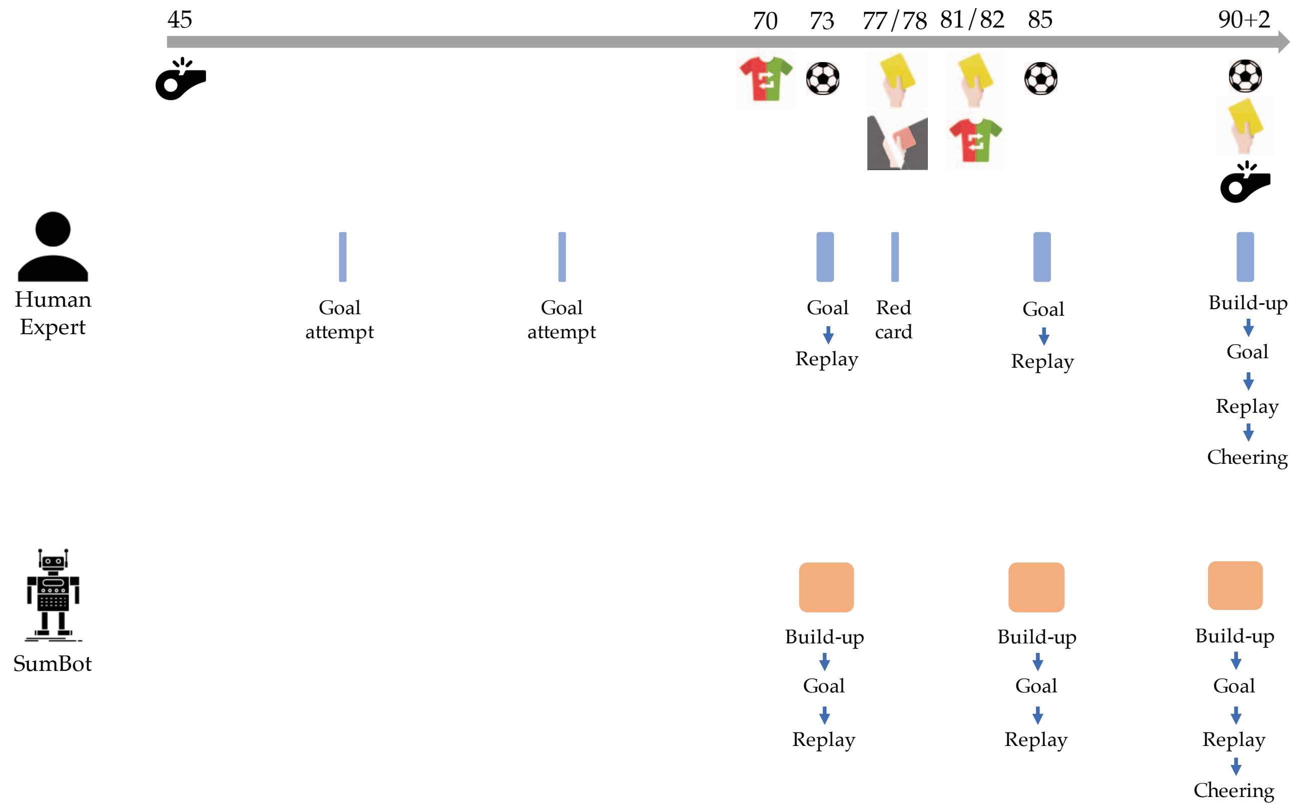
V. Conclusion

A. Summary of our Contributions

In this paper, we have proposed a novel framework for automatic video summarization. The SumBot framework is especially designed for those scenarios where the summarization process follows a semi-structured editorial template. Particularly, we formulate this problem as a markov decision process and employ the max entropy inverse reinforcement learning algorithm to learn the implicit reward function, i.e., the hidden structure/template followed by human experts during the summarization process. Compared to other supervised methods that would require large training sets, our method only requires a small set of expert demonstrations. For a soccer game highlight generation use case, we show that as little as 100 high-quality human-generated highlights are sufficient for our method to learn to mimic the behavior of the human experts. Moreover, quantitative and qualitative results confirm the performance of the proposed approach when compared to other baseline summarization algorithms. To conclude, here we identify a number of research challenges and areas for improvements that will be pursued as future work in this area.



(a)



(b)

Fig. 4: Qualitative visualization of the summary produced by a human expert and our SumBot agent (First half - (a), Second half - (b)). Light blue and light orange segments indicate the portion of the match selected for the summary by the human expert and the SumBot agent, respectively.

B. Open Research Questions

Video shot classification. In Section III-A, a supervised approach is used to obtain the classification labels for each video shot. This semantic information is important because a human expert generally follows a structure to decide what shot to include in the summary. When training labels are not available, we would need to employ an unsupervised approach to retrieve the main types of semantic shots that can compose a highlight, for a given summarization domain. We can leverage that, in some domains, similar shots can appear multiple times in the same highlight video, and these shots are commonly seen across all the video summaries within the same domain.

Transfer learning. Even though our method does not need a large number of training data, there might be summarization use cases where we do not have access to enough high-quality expert demonstrations. A possible solution would be to use a transfer learning approach, where the SumBot agent trained for domain A can be leveraged to initialize the summarization task on domain B, assuming that that the two domains share some high-level characteristics. As an example, some sport domains, albeit different among each other, share some high-level similarities (e.g., hockey vs field hockey vs handball). The agent trained on a certain domain A could therefore be leveraged to initialize the learning for the target domain B, provided that a high-level mapping between the editing templates of A and B is possible. The additional trajectories from the target domain B can then be used to fine-tune and further specialize the reward function learned by the agent.

References

[1] K. M. Mahmoud, M. A. Ismail, and N. M. Ghanem, "Vscan: an enhanced video summarization using density-based spatial clustering," in *International conference on image analysis and processing*. Springer, 2013, pp. 733–742.

[2] B. Mahasseni, M. Lam, and S. Todorovic, "Unsupervised video summarization with adversarial lstm networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 202–211.

[3] H. Gu and V. Swaminathan, "From thumbnails to summaries-a single deep neural network to rule them all," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2018, pp. 1–6.

[4] P. Mundur, Y. Rao, and Y. Yesha, "Keyframe-based video summarization using delaunay clustering," *International Journal on Digital Libraries*, vol. 6, no. 2, pp. 219–232, 2006.

[5] S. E. de Avila, A. da_Luz Jr, A. d. A. Araújo, and M. Cord, "Vsumm: An approach for automatic video summarization and quantitative evaluation," in *2008 XXI Brazilian Symposium on Computer Graphics and Image Processing*. IEEE, 2008, pp. 103–110.

[6] M. Furini, F. Geraci, M. Montangero, and M. Pellegrini, "Stimo: Still and moving video storyboard for the web scenario," *Multimedia Tools and Applications*, vol. 46, no. 1, p. 47, 2010.

[9] Y. Jung, D. Cho, D. Kim, S. Woo, and I. S. Kweon, "Discriminative feature learning for unsupervised video summarization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8537–8544.

[7] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, "Video summarization with long short-term memory," in *European conference on computer vision*. Springer, 2016, pp. 766–782.

[8] K. Zhou, Y. Qiao, and T. Xiang, "Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward," *arXiv preprint arXiv:1801.00054*, 2017.

[10] T. Yao, T. Mei, and Y. Rui, "Highlight detection with pairwise deep ranking for first-person video summarization," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 982–990.

[11] P. Gunawardena, O. Amila, H. Sudarshana, R. Nawaratne, A. Luhach, D. Alahakoon, A. S. Perera, C. Chitraranjan, N. Chilamkurti, and D. De Silva, "Real-time automated video highlight generation with dual-stream hierarchical growing self-organizing maps," *In Journal of Real-Time Image Processing*, 2020.

[12] B. Xiong, Y. Kalantidis, D. Ghadiyaram, and K. Grauman, "Less is more: Learning highlight detection from video duration," in *In Proceedings of the IEEEConference on Computer Vision and Pattern Recognition*, 2019, pp. 1258–1267.

[13] H. Kim, T. Mei, H. Byun, and T. Yao, "Exploiting web images for video highlight detection with triplet deep ranking," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2415–2426, 2018.

[14] Y. Jiao, T. Zhang, S. Huang, B. Liu, and C. Xu, "Video highlight detection via region-based deep ranking model," *In International Journal of Pattern Recognition and Artificial Intelligence*, vol. 33, no. 7, 2019.

[15] S. Lan, R. Panda, Q. Zhu, and A. K. Roy-Chowdhury, "Ffnet: Video fast-forwarding via reinforcement learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6771–6780.

[16] Y. Chen, L. Tao, X. Wang, and T. Yamasaki, "Weakly supervised video summarization by hierarchical reinforcement learning," in *Proceedings of the ACM Multimedia Asia*, 2019, pp. 1–6.

[17] L. Wang, Y. Zhu, and H. Pan, "Unsupervised reinforcement learning for video summarization reward function," in *Proceedings of the 2019 International Conference on Image, Video and Signal Processing*, 2019, pp. 40–44.

[18] N. Krawetz, "Difference hash (dhash) algorithm," http://www.hackerfactor.com/blog/index.php?/archives/529-Kind-of-Like-That.html, 2013.

[19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[21] Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," in *Advances in neural information processing systems*, 2016, pp. 892–900.

[22] A. Secareanu, "Football events," http://https://www.kaggle.com/secareanualin/football-events.

[23] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[24] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum entropy deep inverse reinforcement learning," *arXiv preprint arXiv:1507.04888*, 2015.

[25] A. Ng, "Reinforcement learning and control," http://cs229.stanford.edu/notes/cs229-notes12.pdf.

[26] K. Zhou, T. Xiang, and A. Cavallaro, "Video summarisation by classification with deep reinforcement learning," *arXiv preprint arXiv:1807.03089*, 2018.

[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *In NIPS Workshop*, 2013.