Video Summarization Using AI

Review Report- 2

1st November 2024 - 2nt December 2024

Faculty Mentors:

Prof. Prasad H B prasadhb@pes.edu
Dr. Gowri Srinivasa gsrinivasa@pes.edu
Prof. Preet Kanwal preetkanwal@pes.edu

Video Summarization - Proposed Framework

The video summarization process for sports, such as cricket or soccer, involves extracting key events and highlights from raw footage to create concise summaries. This is achieved through a combination of video content analysis, audio detection, and textual extraction, followed by data preprocessing and summarization. Following are the most important points to be considered while creating the workflow for such a problem :

Input Analysis:

- **Video Content Analysis**:
  - **Scene/Event Detection**: Identify meaningful scenes (e.g., celebrations, wickets, or milestones in cricket) using event detection algorithms and object recognition techniques.
  - **Key Frame Extraction**: Extract representative frames from videos, focusing on highlights like wickets or boundaries.
  - **Player and Ball Tracking**: Recognize players and track the ball to enhance event detection.
- **Audio Content Analysis**:
  - Detect sound cues such as clapping, cheering, or commentary highlights.
- **Textual Content Extraction**:
  - Use Optical Character Recognition (OCR) for text embedded in the video, like score overlays.
  - Extract metadata like match specifics, overs, and other relevant annotations.

Data Representation and Preprocessing:

- Utilize models like BERT or other domain-specific embeddings to structure data, enabling seamless representation across innings, overs, or events.
- Align textual, visual, and auditory content for consistency.

Summarization Engine:

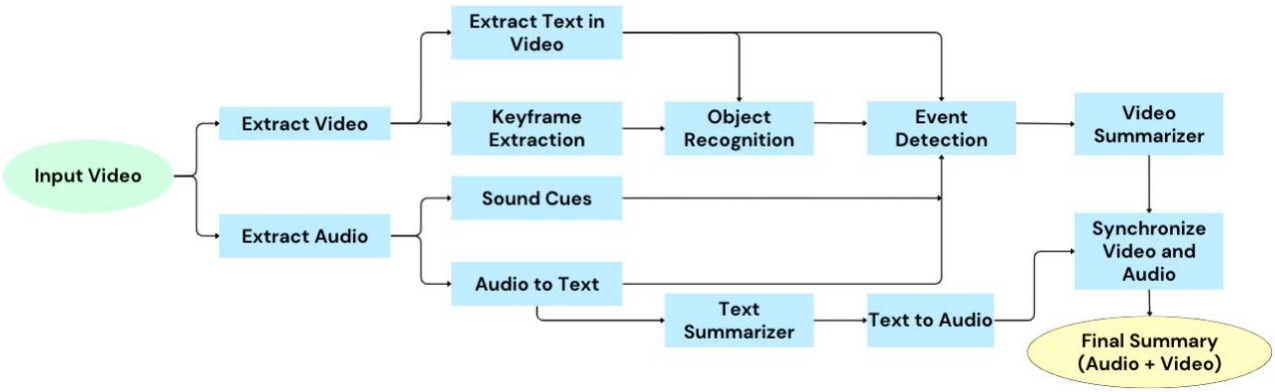
- **Content Summarizer**:
  - Text Summarization: Use advanced NLP techniques to compress commentary or match descriptions.
  - Audio-Visual Stitching: Compile relevant frames and audio snippets for output.
- **Highlight Detection**:
  - Event Detection: Automatically identify significant moments (boundaries, wickets, fielding feats).
  - Milestone Tracking: Detect milestones (e.g., 50 runs, maiden overs).
- **Customization**:
  - Team-Specific Highlights: Generate content tailored to teams or players.
  - Audience Preferences: Adjust summarization based on user queries.

Post-Production:

- Combine processed video, audio, and textual summaries into cohesive snippets.
- Render video output in desired formats (mp4, avi).

Output Generation:

- Compile summarized videos with metadata for easy search and access.
- Integrate playback options for specific events or timelines.



The diagram illustrates a modular pipeline for video summarization, systematically processing video and audio components to generate a concise summary. Below is a detailed explanation of each module in the workflow, along with methods and techniques that can be applied.

1. Input Video

This is the raw video file that serves as the source material for summarization. The pipeline takes the video as input and separates it into two streams: video and audio, for specialized processing.

2. Extract Video

- **Purpose**: Separate the visual content from the input video for analysis.
- **Methods**:
  - **Frame Decoding**: Extract frames from the video using libraries like OpenCV, FFmpeg, or PyAV.
  - **Preprocessing**: Resize and normalize frames to prepare them for further analysis (e.g., downscaling to reduce computational load).

3. Keyframe Extraction

- **Purpose**: Select the most representative frames that encapsulate the essence of the video.
- **Methods**:
    - **Histogram-Based Approaches**: Analyze color histograms across frames to detect scene changes.
    - **Motion Analysis**: Use optical flow to identify frames with significant movement.
    - **Deep Learning Models**:
        - CNN-based autoencoders for redundancy reduction.
        - Clustering algorithms (e.g., K-means) to group similar frames and choose representative ones.
    - **Applications**:
        - Reduce the number of frames for event detection.
        - Focus on frames containing relevant objects.
- **Example :** Extract frames from moments when goals are scored, or when fouls, offside violations, or player injuries occur.

4. Object Recognition

- **Purpose**: Identify and detect objects within video frames to provide context and prioritize important content.
- **Methods**:
    - **Traditional Techniques**: Feature-based object recognition using HOG or SIFT for basic detection.
    - **Deep Learning Approaches**:
        - **YOLO (You Only Look Once)**: Real-time object detection for detecting multiple objects in a single pass.
        - **Faster R-CNN**: High-accuracy region-based object detection.
        - **Mask R-CNN**: Object detection with pixel-level segmentation.
    - **Applications**:
        - Detecting objects like players, vehicles, or items of interest.
        - Tracking object movements across frames for temporal analysis.
- **Example** : Detect players, the soccer ball, and the goalposts. Recognize which player is in possession of the ball or involved in a key play (like a goal or assist).

5. Event Detection

- **Purpose**: Identify significant events in the video, such as goals, milestones, or key actions.
- **Methods**:
    - **Rule-Based Techniques**:
        - Domain-specific rules (e.g., detecting ball crossing the boundary in cricket).
    - **Machine Learning**:
        - Spatiotemporal feature extraction using LSTMs or RNNs to model event sequences.
        - 3D-CNNs for detecting events involving both spatial and temporal changes.
    - **Applications**:
        - Extract highlights of sports matches or key scenes in movies.
        - Correlate object interactions (e.g., players interacting with a ball).
- **Example** : Detect goals by recognizing when the ball enters the goalpost area, and identify fouls, penalties, or offside events.

6. Extract Audio

- **Purpose**: Separate the audio content from the input video for analysis.
- **Methods**:
    - **Audio Feature Extraction**: Use tools like Librosa to analyze audio waveforms for pitch, amplitude, and frequency.
    - **Noise Reduction**: Filter out background noise to enhance clarity.

7. Sound Cues

- **Purpose**: Identify meaningful audio patterns or sound effects.
- **Methods**:
    - **Spectrogram Analysis**: Convert audio signals into spectrograms for identifying patterns like clapping or cheering.
    - **Deep Learning**:
        - Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs) for audio classification.
    - **Applications**:
        - Detecting audience reactions or critical audio events (e.g., a referee's whistle).

8. Audio to Text

- **Purpose**: Transcribe spoken audio content (e.g., commentary) into text for further processing.
- **Methods**:
  - **Automatic Speech Recognition (ASR)**:
    - Open-source tools: Google Speech-to-Text API, DeepSpeech, Whisper by OpenAI.
    - Domain-specific fine-tuned models for better accuracy.
  - **Applications**:
    - Extracting context from commentary.
    - Enabling cross-modal analysis by aligning text and video.

9. Text Summarizer

- **Purpose**: Summarize textual content obtained from audio transcription or extracted text.
- **Methods**:
  - **Extractive Summarization**: Select key sentences using algorithms like TextRank or LexRank.
  - **Abstractive Summarization**:
    - Deep learning models like BERT, or T5 fine-tuned for summarization.
  - **Applications**:
    - Provide concise descriptions of events.
    - Add subtitles or text overlays to summarized videos.

10. Text to Audio

- **Purpose**: Convert summarized text back into audio for integration with the video summary.
- **Methods**:
  - **Text-to-Speech (TTS)**:
    - Tools like Google TTS, Amazon Polly, or Tacotron for natural-sounding speech synthesis.
  - **Applications**:
    - Generate commentary or voiceovers for summarized videos.

11. Video Summarizer

- **Purpose**: Combine the results of object recognition, keyframe extraction, and event detection into a coherent visual summary.
- **Methods**:
  - **Timeline Compression**:
    - Condense relevant events into a shorter time span.
  - **Highlight Compilation**:
    - Use editing tools to arrange extracted keyframes and events.
  - **Applications**:
    - Create highlights for sports, lectures, or entertainment.

12. Correlate Video and Audio

- **Purpose**: Align visual and audio components to produce a seamless summary.
- **Methods**:
  - **Temporal Synchronization**:
    - Align video keyframes with audio commentary or sound cues.
  - **Applications**:
    - Ensure smooth transitions and synchronized playback.

13. Final Summary (Audio + Video)

- **Purpose**: Produce the final summarized content combining all processed components.
- **Methods**:
  - **Rendering**:
    - Use video editing libraries like MoviePy or Adobe Premiere APIs.
  - **Meta-Tagging**:
    - Add metadata (e.g., timestamps, events) for easy retrieval.
  - **Applications**:
    - Distribute summarized content for various platforms like social media, streaming services, or reports.

Conclusion

This modular pipeline proposes a set of alternatives for each of the components in the workflow for video summarization. In the next couple of months, we propose to explore these alternatives and provide a feasible proof of concept for summarizing the game of soccer as suggested by the Ittiam Team.