Chopped: Audio Driven Video Segmentation

Matthew Flanagan, Philip Maly, Aidan Grady, Evan Wong - University of New Hampshire

University of New Hampshire
Interoperability Labs

Introduction

High school baseball coaches often spend significant time manually reviewing and editing game footage to analyze player performance. This process is time consuming and inefficient when working with full-length game recordings.

To address this challenge, Chopped is a web application designed to automatically sort and segment game film. The system utilizes multimedia data processing, incorporating both audio and structured data files to identify and organize key moments within a game. The system is hardware-independent, requiring only recorded video and audio input.

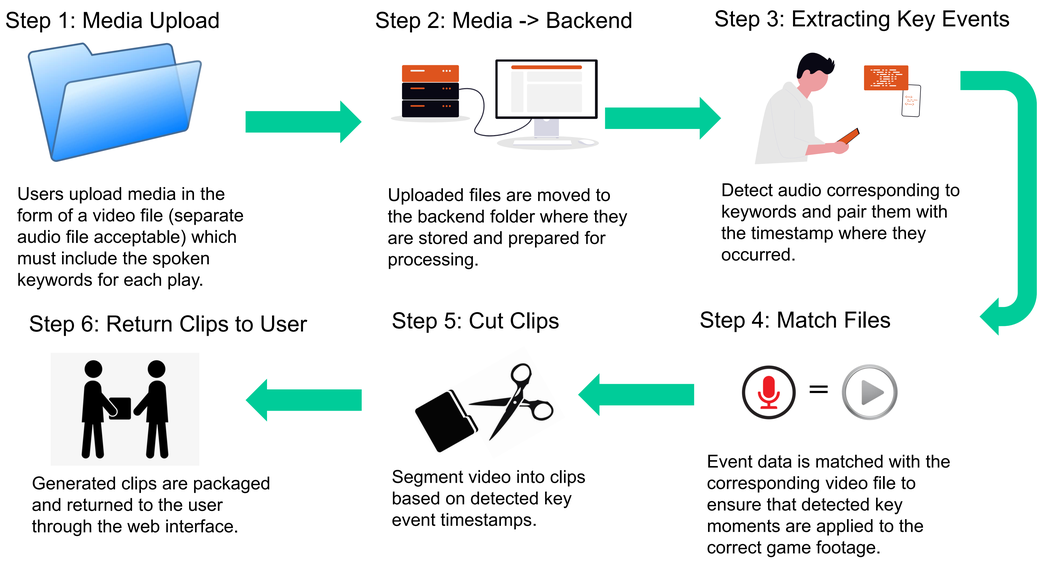
Methods

- Web application developed using Python, HTML, and CSS with a Flask-based backend.
- Utilizes integrated libraries for video processing and speech-to-text transcription.
- Users verbally mark key events during recording (e.g. “strikeout”, “double”, “E6”, etc.).
- Spoken keywords are transcribed and mapped to timestamps.
- Uploaded video and audio files are processed together to generate segmented game footage

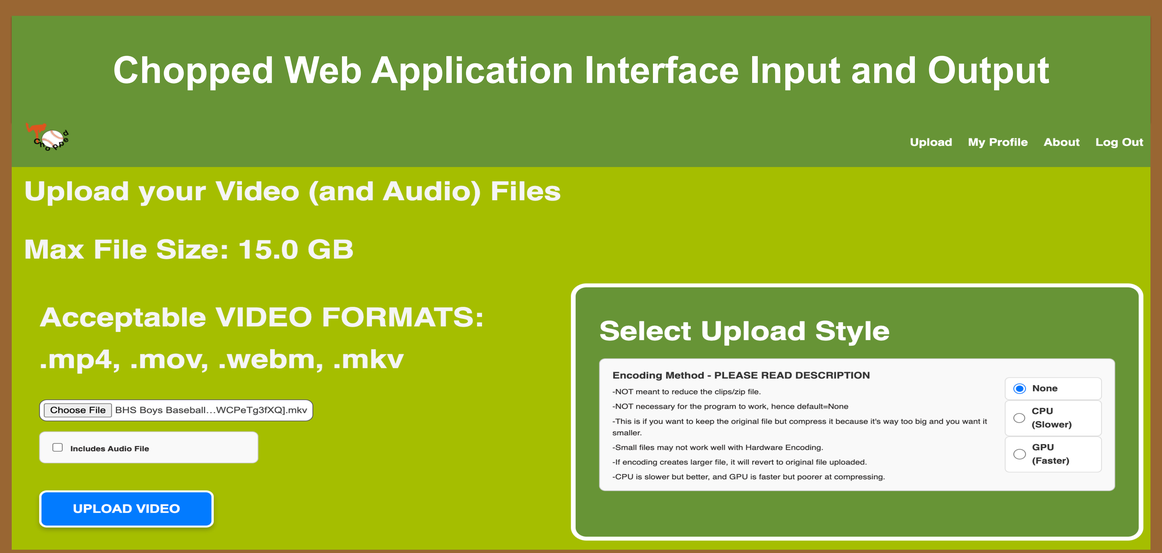
System Metrics/Summary

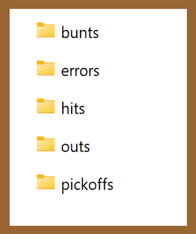
- Input: ~2 hour game recording with the option for an additional audio file
- Output: Multiple organized 25-second video clips of key events
- Processing: Fully automated after user upload
- Accuracy: High when keywords are clearly spoken

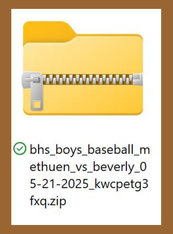
Data Flow



Chopped Web Application Interface Input and Output







Results

- Tested on full-length (~2 hour) baseball game recordings with live announcer audio.
- System successfully detected and generated clips when relevant keywords (e.g. “double”, “strikeout”) were clearly spoken.
- Performance depended on consistency of spoken cues, with extra clips generated when announcers did not follow expected phrasing.
- Simulated ideal usage by manually creating event data, confirming that clips were generated at correct timestamps.
- Background noise had minimal impact, as the system selectively detects target keywords rather than general speech.
- Repeated keywords at different timestamps result in multiple clips, reflecting direct mapping between detected events and generated segments.

Future Steps/Implications

- Transition to a hybrid architecture, where video processing runs locally while a web platform stores and organizes generated clips.
- Improve system robustness to background noise and inconsistent spoken input.
- Enable real-time or near real-time clip generation, accounting for hardware and streaming constraints.
- Expand keyword detection to support a broader range of events and sports.
- Develop an interactive interface for reviewing, searching, and organizing clips.