Detecting Scene Changes in Audiovisual Content

Netflix Technology Blog · Follow
Published in Netflix TechBlog
6 min read · Jun 20, 2023

Listen Share

Avneesh Saluja, Andy Yao, Hossein Taghavi

Introduction

When watching a movie or an episode of a TV show, we experience a cohesive narrative that unfolds before us, often without giving much thought to the underlying structure that makes it all possible. However, movies and episodes are not atomic units, but rather composed of smaller elements such as frames, shots, scenes, sequences, and acts. Understanding these elements and how they relate to each other is crucial for tasks such as video summarization and highlights detection, content-based video retrieval, dubbing quality assessment, and video editing. At Netflix, such workflows are performed hundreds of times a day by many teams around the world, so investing in algorithmically-assisted tooling around content understanding can reap outsized rewards.

While segmentation of more granular units like frames and shot boundaries is either trivial or can primarily rely on pixel-based information, higher order segmentation[1] requires a more nuanced understanding of the content, such as the narrative or emotional arcs. Furthermore, some cues can be better inferred from modalities other than the video, e.g. the screenplay or the audio and dialogue track. Scene boundary detection, in particular, is the task of identifying the transitions between scenes, where a scene is defined as a continuous sequence of shots that take place in the same time and location (often with a relatively static set of characters) and share a common action or theme.

In this blog post, we present two complementary approaches to scene boundary detection in audiovisual content. The first method, which can be seen as a form of weak supervision, leverages auxiliary data in the form of a screenplay by aligning screenplay text with timed text (closed captions, audio descriptions) and assigning timestamps to the screenplay's scene headers (a.k.a. sluglines). In the second approach, we show that a relatively simple, supervised sequential model (bidirectional LSTM or GRU) that uses rich, pretrained shot-level embeddings can outperform the current state-of-the-art baselines on our internal benchmarks.

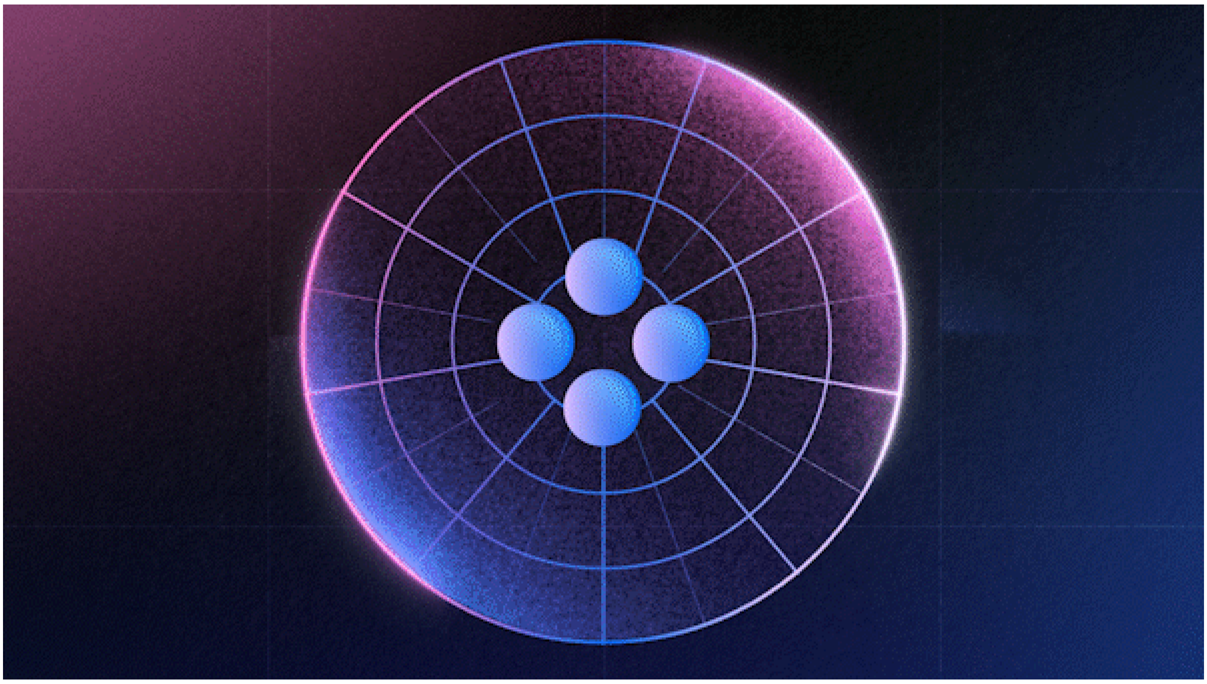
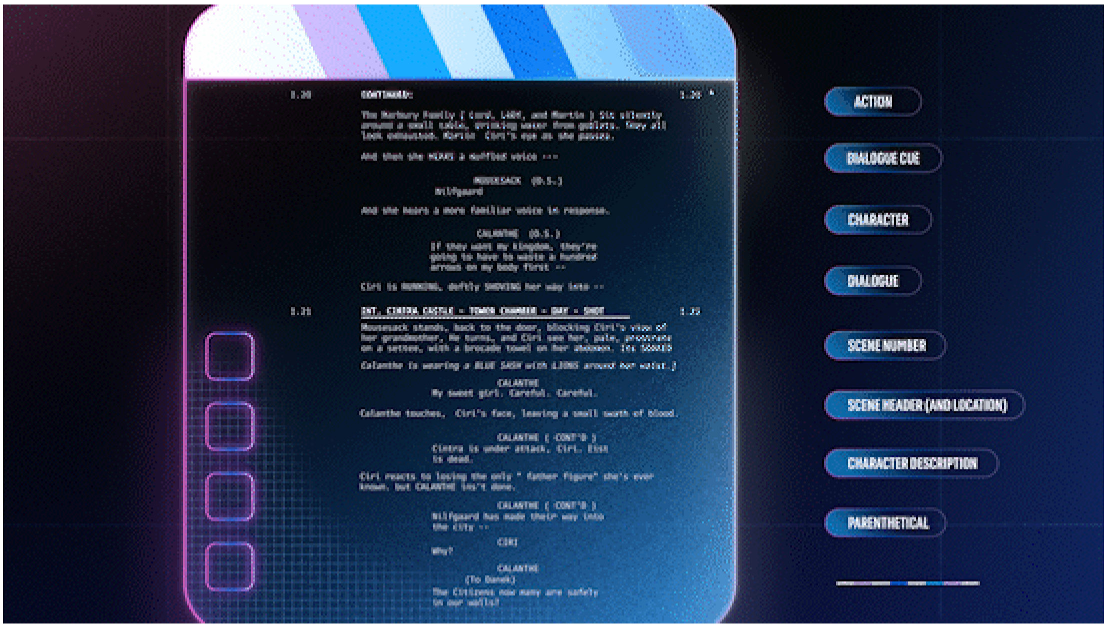


Figure 1: a scene consists of a sequence of shots.

Leveraging Aligned Screenplay Information

Screenplays are the blueprints of a movie or show. They are formatted in a specific way, with each scene beginning with a scene header, indicating attributes such as the location and time of day. This consistent formatting makes it possible to parse screenplays into a structured format. At the same time, a) changes made on the fly (directorial or actor discretion) or b) in post production and editing are rarely reflected in the screenplay, i.e. it isn't rewritten to reflect the changes.



1.20 CONTINUED: 1.20 *

The Marbury Family (Lord, Lady, and Martin) Sit silently around a small table, drinking water from goblets. They all look exhausted. Martin Ciri's eye as she passes.

And then she HEARS a muffled voice --

MOUSESACK (O.S.)
Nilfgaard

And she hears a more familiar voice in response.

CALANTHE (O.S.)
If they want my kingdom, they're going to have to waste a hundred arrows on my body first --

Ciri is RUNNING, deftly SHOVING her way into --

1.21 INT. CINTRA CASTLE - TOWER CHAMBER - DAY - SHOT 1.21

Mousesack stands, back to the door, blocking Ciri's view of her grandmother. He turns, and Ciri see her, pale, prostrate on a settee, with a brocade towel on her abdomen. Its SOAKED

Calanthe is wearing a BLUE SASH with LIONS around her waist.)

CALANTHE
My sweet girl. Careful. Careful.

Calanthe touches, Ciri's face, leaving a small swath of blood.

CALANTHE (CONT'D)
Cintra is under attack, Ciri. Eist is dead.

Ciri reacts to losing the only " father figure" she's ever known, but CALANTHE isn't done.

CALANTHE (CONT'D)
Nilfgaard has made their way into the city --

CIRI
Why?

CALANTHE
(To Danek)
The Citizens now many are safely in our walls?

Figure 2: screenplay elements, from The Witcher S1E1.

In order to leverage this noisily aligned data source, we need to align time-stamped text (e.g. closed captions and audio descriptions) with screenplay text (dialogue and action[2] lines), bearing in mind a) the on-the-fly changes that might result in semantically similar but not identical line pairs and b) the possible post-shoot changes that are more significant (reordering, removing, or inserting entire scenes). To address the first challenge, we use pre trained sentence-level embeddings, e.g. from an embedding model optimized for paraphrase identification, to represent text in both sources. For the second challenge, we use dynamic time warping (DTW), a method for measuring the similarity between two sequences that may vary in time or speed. While DTW assumes a monotonicity condition on the alignments[3] which is frequently violated in practice, it is robust enough to recover from local misalignments and the vast majority of salient events (like scene boundaries) are well-aligned.

As a result of DTW, the scene headers have timestamps that can indicate possible scene boundaries in the video. The alignments can also be used to e.g., augment audiovisual ML models with screenplay information like scene-level embeddings, or transfer labels assigned to audiovisual content to train screenplay prediction models.

Figure 3: alignments between screenplay and video via time stamped text for The Witcher S1E1.

A Multimodal Sequential Model

The alignment method above is a great way to get up and running with the scene change task since it combines easy-to-use pretrained embeddings with a well-known dynamic programming technique. However, it presupposes the availability of high-quality screenplays. A complementary approach (which in fact, can use the above alignments as a feature) that we present next is to train a sequence model on annotated scene change data. Certain workflows in Netflix capture this information, and that is our primary data source; publicly-released datasets are also available.

From an architectural perspective, the model is relatively simple — a bidirectional GRU (biGRU) that ingests shot representations at each step and predicts if a shot is at the end of a scene.[4] The richness in the model comes from these pretrained, multimodal shot embeddings, a preferable design choice in our setting given the difficulty in obtaining labeled scene change data and the relatively larger scale at which we can pretrain various embedding models for shots.

For video embeddings, we leverage an in-house model pretrained on aligned video clips paired with text (the aforementioned "timestamped text"). For audio embeddings, we first perform source separation to try and separate foreground (speech) from background (music, sound effects, noise), embed each separated waveform separately using wav2vec2, and then concatenate the results. Both early and late-stage fusion approaches are explored; in the former (Figure 4a), the audio and video embeddings are concatenated and fed into a single biGRU, and in the latter (Figure 4b) each input modality is encoded with its own biGRU, after which the hidden states are concatenated prior to the output layer.

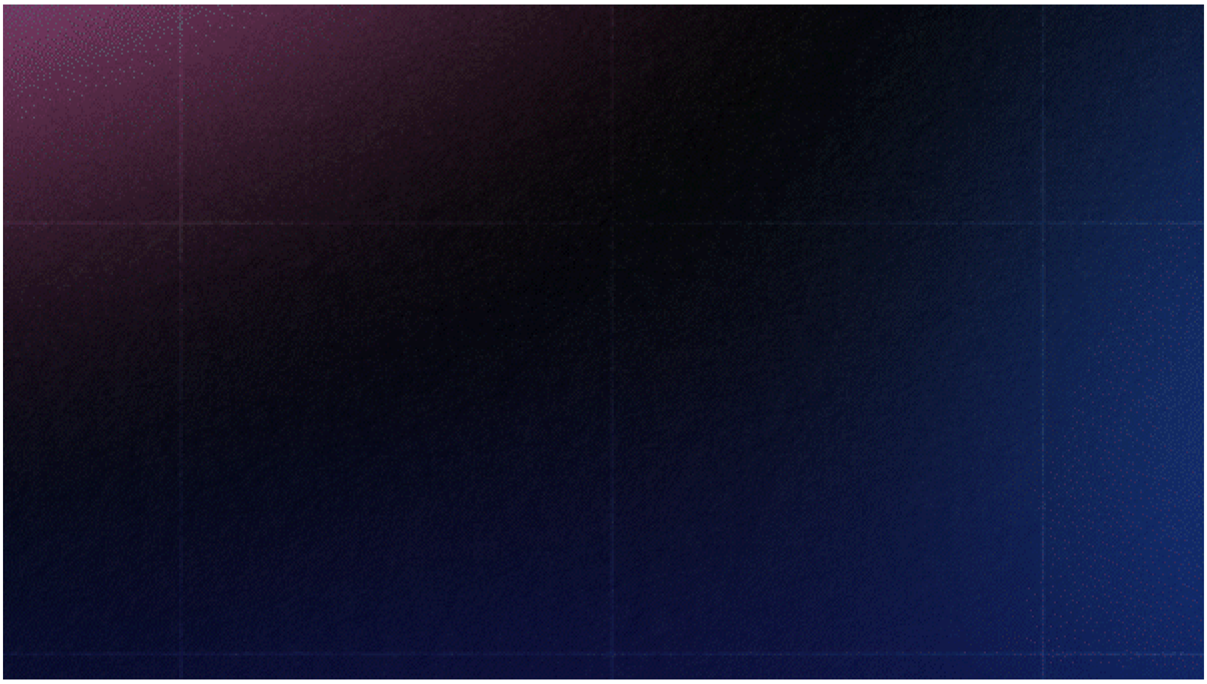


Figure 4a: Early Fusion (concatenate embeddings at the input).

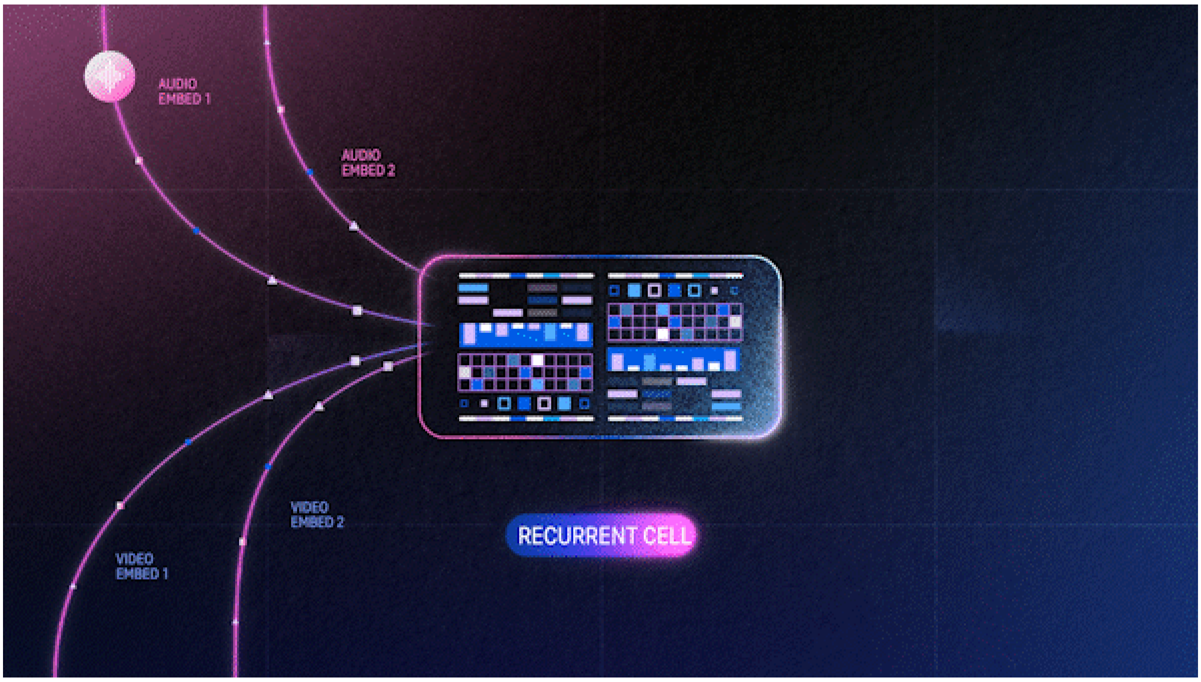


Figure 4b: Late Fusion (concatenate prior to prediction output).

We find:

- Our results match and sometimes even outperform the state-of-the-art (benchmarked using the video modality only and on our evaluation data). We evaluate the outputs using F-1 score for the positive label, and also relax this evaluation to consider "off-by-*n*" F-1 i.e., if the model predicts scene changes within *n* shots of the ground truth. This is a more realistic measure for our use cases due to the human-in-the-loop setting that these models are deployed in.

- As with previous work, adding audio features improves results by 10–15%. A primary driver of variation in performance is late vs. early fusion.
- Late fusion is consistently 3–7% better than early fusion. Intuitively, this result makes sense — the temporal dependencies between shots is likely modality-specific and should be encoded separately.

Conclusion

We have presented two complementary approaches to scene boundary detection that leverage a variety of available modalities — screenplay, audio, and video. Logically, the next steps are to a) combine these approaches and use screenplay features in a unified model and b) generalize the outputs across multiple shot-level inference tasks, e.g. shot type classification and memorable moments identification, as we hypothesize that this path would be useful for training general purpose video understanding models of longer-form content. Longer-form content also contains more complex narrative structure, and we envision this work as the first in a series of projects that aim to better integrate narrative understanding in our multimodal machine learning models.

Special thanks to Amir Ziai, Anna Pulido, and Angie Pollema.

Footnotes

1. Sometimes referred to as boundary detection to avoid confusion with image segmentation techniques.
2. Descriptive (non-dialogue) lines that describe the salient aspects of a scene.

4. We experiment with adding a Conditional Random Field (CRF) layer on top to enforce some notion of global consistency, but found it did not improve the results noticeably.

Machine Learning Computer Vision Multimodal NLP

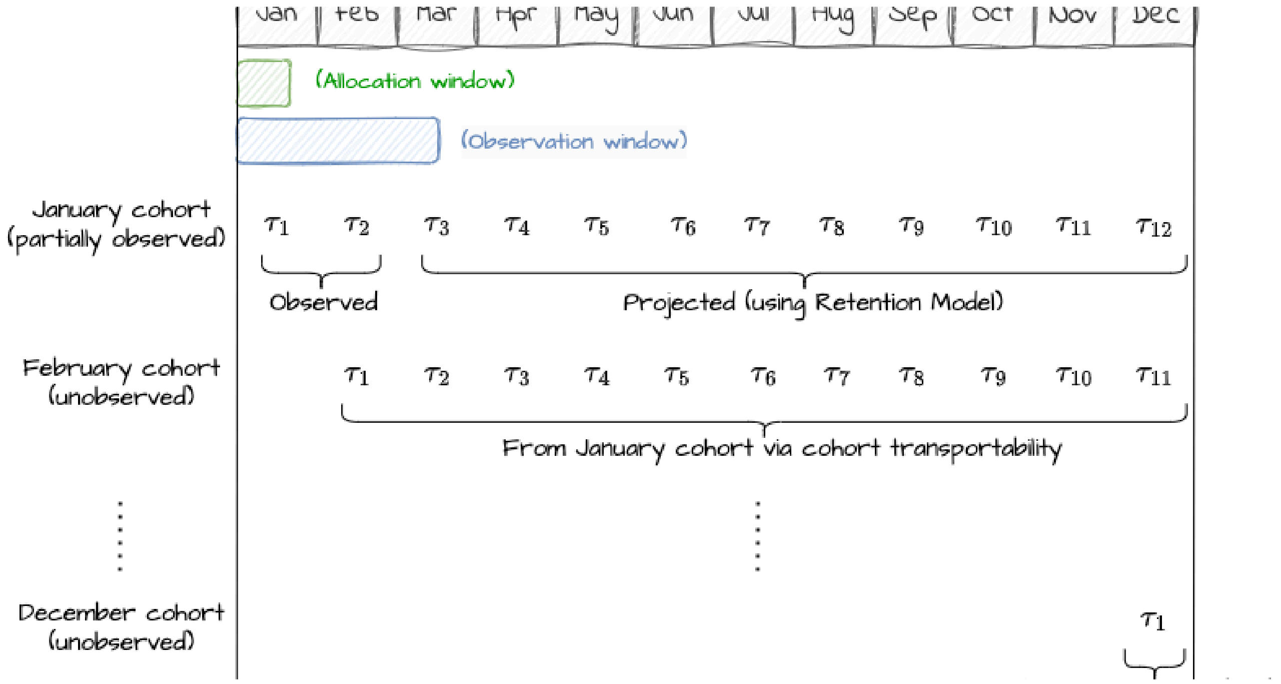
Follow

Written by Netflix Technology Blog

412K Followers · Editor for Netflix TechBlog

Learn more about how Netflix designs, builds, and operates our systems and engineering organizations

More from Netflix Technology Blog and Netflix TechBlog


Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
(Allocation window)
(Observation window)
January cohort (partially observed)
T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11 T12
Observed
Projected (using Retention Model)
February cohort (unobserved)
T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11
From January cohort via cohort transportability
December cohort (unobserved)
T1