Revolutionizing TV Show Experience: Using Recaps for Multimodal Story Summarization

Thesis submitted in partial fulfilment
of the requirements for the degree of

Master of Science in ***Computer Science and Engineering*** *by Research*

by

Aditya Kumar Singh
2021701010
aditya.si@research.iiit.ac.in
Advisor: Prof. Makarand Tapaswi

International Institute of Information Technology
Hyderabad - 500 032, INDIA
April 2024

International Institute of Information Technology

Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Revolutionizing TV Show Experience: Using Recaps for Multimodal Story Summarization” by Aditya Kumar Singh, has been carried out under my supervision and is not submitted elsewhere for a degree.

April 2024

Advisor: Prof. Makarand Tapaswi

To my Friends, Family, and Indian Military

Acknowledgments

In the vast landscape of academic pursuits, this thesis stands as a testament to the dedication, perseverance, and collaboration that have shaped its creation. Reflecting on the journey of the past two and a half years, I am profoundly grateful for the many hands that guided me, the hearts that supported me, and the minds that inspired me. This transformative period has not only influenced my research perspective but has also bestowed upon me countless invaluable memories.

First and foremost, I extend my deepest appreciation and **HUGE** thanks to my thesis advisor, Prof. Makarand Tapaswi. One significant aspect that I've observed, partially learned, feel fortunate to have, and will be forever indebted for, is his guidance, perseverance, leadership, and overall personality. Thank you for generously investing your time and patience in listening to my most trivial doubts and engaging in discussions about them. Thank you for allowing ample time for contemplation and for collaborating on plans. Your understanding of my challenges and proactive measures to address them have been invaluable. Your support has not only been instrumental in my thesis work but has also propelled countless other endeavors to fruition.

Now, I wish to express my profound and heartfelt gratitude to my friend, lab-mate, and project-partner, *Dhruv*. I look up to him before making any significant decision, whether it's professional or personal. His immense assistance and expertise in coding have elevated my skills from nearly zero to HERO. Just as having a partner in a marathon makes the obstacles feel conquerable, the same can be said of him. With individuals like him, the *research marathon* also seems attainable.

Coming up next, a **BIGGG** shoutout to my IIIT-H mini-family – *Dhruv*, *Bhoomeendra*, *Dhaval*, *Nayan*, *Amruth*, and *Prateek* (Our *Chicha* – akin to a father figure). From the moment I joined this college, this group spontaneously formed (on WhatsApp; not promoting), which in itself is an interesting story (*Drake* sings *GOD's Plan*), and it's bound to last forever. Throughout my tenure, I assumed various leadership roles (*e.g.*, Parliament, Hostel, Placement, *etc.*), where I may have been the "frontend", but the "backend" was always *we*. Your companionship made challenges seem surmountable and victories all the sweeter. There are numerous fantastic and engaging stories among us, too many for this thesis to accommodate. Summing it up in one line, we embarked on a multitude of amazing, breathtaking, and wondrous activities (the valid ones, of course ;)), which leaves me with a profound sense of satisfaction and completeness in defining how a graduation life should be. I cannot find enough words to acknowledge you all properly, so I'll simply say, "Thank You!" and dedicate my thesis to you guys.

Now is the time for a proper *army-style* **HATS-OFF** followd by a gentle HUG to my constant supporters, my lifeline, my emotional rescue-team, and my creators – *Baba*, *Maa*, and my brother, *Sandeep*. They are the reason why I am here and why I strive for more, dream bigger, and persist. Your boundless love, unwavering encouragement, and endless sacrifices have been the bedrock of my success, and I dedicate this thesis to you with all my heart.

Last but certainly not least, to the countless unnamed individuals whose contributions, large and small, have shaped my academic journey, thank you for your kindness, generosity, and unyielding belief in my potential. Your support has been the wind beneath my wings, propelling me ever closer to my goals. To my whole IIIT-H family, who, alike its *symbol*, showered with me support, knowledge, and a life worth remembering.

As I stand on the brink of this academic milestone, I am overwhelmed with profound gratitude and humility. This thesis is a reflection of the collective efforts of all those who have touched my life, and it is with the deepest sincerity that I extend my heartfelt thanks to each and every one of you.

Abstract

We introduce a novel approach for *multimodal story summarization*, aimed at leveraging TV episode recaps to create concise summaries of complex storylines. These recaps, which consist of short video sequences combining key visual moments and dialogues from previous episodes, serve as a valuable source of weak supervision for labeling the summarization task.

To facilitate this approach, we introduce the *PlotSnap* dataset, which focuses on two crime thriller TV shows. Each episode in this dataset is over 40 minutes long and is accompanied by rich recaps. These recaps are mapped to corresponding sub-stories, providing labels for the story summarization task.

Our proposed model, *TaleSumm*, operates hierarchically.

(i) First, it processes entire episodes by generating compact representations of shots and dialogues.

(ii) Then, it predicts the importance scores for each video shot and dialog utterance, taking into account interactions between local story groups.

Unlike traditional summarization tasks, our method extracts multiple plot points from long-form videos. We conducted a comprehensive evaluation of our approach, including assessing its performance in cross-series generalization. TaleSumm demonstrates promising results, not only on the video summarization benchmarks but also in effectively summarizing the intricate storylines of the TV shows in the PlotSnap dataset. Our project implementation as well as dataset features and demo can be found at `https://github.com/katha-ai/RecapStorySumm-CVPR2024`.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Imagine settling in for your favorite TV series, eager to catch up on the latest episode. You hit play, and there it is, the familiar "*Previously on...*" —the recap. It's that quick cinematic journey that swiftly brings you up to speed, reminding you of key moments from past episodes. But have you ever noticed the spellbinding charm of these recaps, how elegantly they interweave snippets of the story with just the right bits of dialogue to spark your memory and set the stage for what's next?

In today's streaming era of binge-watching marathons, recaps continue to be our faithful guide, ensuring we're always aware of the labyrinth of intricate storylines. What if we could harness the power of recaps not just to jog your memory but to fuel the creation of compelling story summaries? That's precisely what we're here to explore. For the following seven pages, we'll embark on a quest where TV storytelling meets modern technology to uncover the untapped potential of recaps for story summarization.

A TV show **recap** is a concise, under-two-minute sequence of crucial plot points from previous episodes. To satisfy the time constraint, the recap is constructed by editing previous episode shots with sharp and rapid cuts, sometimes adding shots from previously unseen footage, and selecting/modifying dialog utterances to ensure relevance to the sub-story. A good recap sets the stage for the main part of the episode by weaving visual and dialog cues to *spark the viewers' memory*. Thus, a recap is a great way to identify sub-stories important to the overall story arc.

Extending this idea, we use recaps to create **story summaries** by identifying and expanding the sub-stories from the episode aiming for a more detailed video understanding, (Fig. 1.1). For identification, we introduce an innovative shot-matching algorithm (Section 2.1) that associates tampered shots from recap to their corresponding shots in the episode.

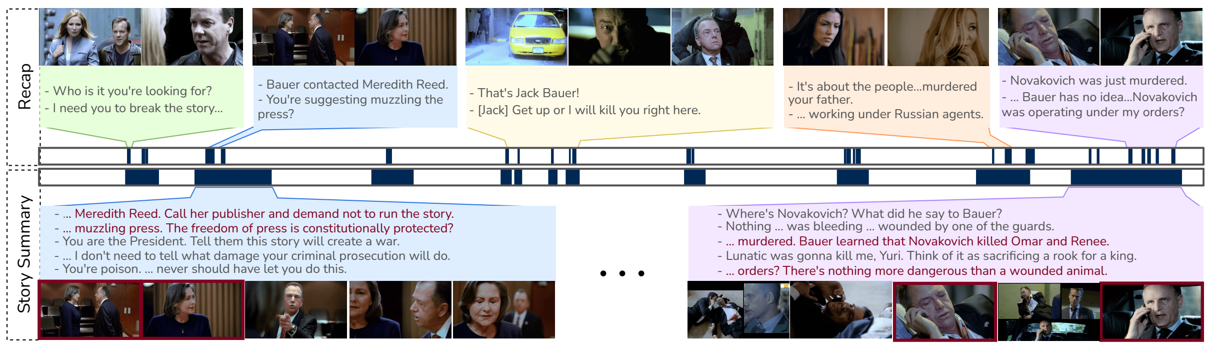
Recap
- Who is it you're looking for?
- I need you to break the story...
- Bauer contacted Meredith Reed.
- You're suggesting muzzling the press?
- That's Jack Bauer!
- [Jack] Get up or I will kill you right here.
- It's about the people...murdered your father.
- ... working under Russian agents.
- Novakovich was just murdered.
- ... Bauer has no idea...Novakovich was operating under my orders?
Story Summary
- ... Meredith Reed. Call her publisher and demand not to run the story.
- ... muzzling press. The freedom of press is constitutionally protected?
- You are the President. Tell them this story will create a war.
- ... I don't need to tell what damage your criminal prosecution will do.
- You're poison. ... never should have let you do this.
. . .
- Where's Novakovich? What did he say to Bauer?
- Nothing ... was bleeding ... wounded by one of the guards.
- ... murdered. Bauer learned that Novakovich killed Omar and Renee.
- Lunatic was gonna kill me, Yuri. Think of it as sacrificing a rook for a king.
- ... orders? There's nothing more dangerous than a wounded animal.

Figure 1.1: We illustrate how TV show recaps can be used to generate labels for *multimodal story summarization*. The *top half* identifies key moments (shots and dialogs) from S08E22 of *24* that feature in the recap at the beginning of the next episode (S08E23). As recaps help viewers recall essential story events, we extend these segments to create summarization labels (visualized in the *bottom half* where shots and dialogs inherited from recap are marked in deep red). For example, in the sub-story (left), the recap hints at Jack Bauer relaying classified information to the press, while the summary presents the complete sub-story, including Logan informing President Taylor about their failure to catch Jack and their disagreement over muzzling the press.

Different from a recap, a story summary consists of entire scenes or sub-stories that are essential to the narrative. They serve as the anchors that keep us grounded in the narrative. They distill the essence of an episode, capturing its pivotal moments and essential elements. Most importantly, it guides viewers seeking a quick yet comprehensive understanding of a drama. Thus, a first-time viewer may watch story summaries of each episode serially and understand the main narrative, while watching recaps serially does not help as they only trigger memories assuming that the viewer has seen the episode before.

Continuing with the same spirit, we bring you two popular crime thrillers, TV shows: *24*and *Prison Break*, marking the introduction of a new dataset, *PlotSnap*, that features several episodes from these shows. Both series fill the moment with suspense and intrigue. On one side, the clock is ticking in the heart-pounding world of *24*, where Jack Bauer relentlessly tackles every seemingly impossible mission. On the other, Michael Scofield in *Prison Break*teases daring escapes and high-stakes adventures. With excellent recaps in both of these shows that serves as *free annotation* for us, we can extract important narrative sub-plots from the recap to create story summaries (see Chapter 2). Despite lasting only 1-2 minutes, we present an innovative strategy for extracting valuable story summary labels from them (see Chapter 2).

We *propose* a novel task of creating multimodal story summaries for TV episodes. Unlike previous multimodal summarization methods [6, 7, 8] that can only generate either a video or a text summary even after leveraging additional modalities, our task of story summarization is an instance of multimodal long-video understanding where an entire episode (typically 40 minutes) needs to be processed. We formulate story summarization as an extractive multimodal summarization with multimodal output (video-text-2-video-text, VT2VT) which further expands our horizon on harnessing the complementary benefits in the additional modality. Specifically, with video shots and dialog utterances (story elements) in the episode as input, we intend to build models that predict scores indicating the importance of each shot *and* utterance. Note, selecting multiple important and connected sub-stories is different and challenging from most works that promote diversity [9].

1.1 Motivation & Contribution

Notably, existing video summarization techniques, in general, aim at extracting "point" summaries, e.g., pinpointing specific actions in instructional videos [7] or prioritizing diversity [9]. In contrast, story summarization emphasizes narrative-centric content, allowing it to retrieve multiple story segments harmoniously aligned with the overarching story arc.

Recently, several studies [10, 11, 6] have explored multimodal summarization with multimodal output (MSMO), which aims to generate video and text summaries using a joint model. However, their limitation (except A2Summm [12]) in establishing temporal correspondence across complementing modalities needs to be exploited. E.g., No existing method utilizes the mutual temporal alignment between a video and its transcripts, which are automatically synchronized. Instead, the two modalities are treated separately. We address this by proposing a hierarchical attention-based model, *TaleSumm* (Chapter 3), that employs a two-tiered strategy for feature capture. As mentioned, A2Summ [12] too, generates both outputs; but we differ significantly in video type (stories *vs*. creative videos), the duration of the input video, and the model architecture. The first level of our model encodes shot and utterance representations. At the second level, we foster interaction between shots and utterances within local story groups based on a temporal neighborhood, reducing the impact of distant and potentially noisy elements. To maintain the narrative's temporal flow, a dedicated group token enables message-passing across story groups. Our model leverages pre-extracted features from established backbones, allowing it to capture interactions across all video shots and dialog utterances within a 40-minute episode. Notably, it maintains efficiency, making it suitable for training on a modest GPU with *12GB* of memory. Plus, we tried summarizing an entire TV episode (42 minutes long on average), which presents a formidable challenge for models accustomed to processing shorter clips lasting just a few seconds to a few minutes.

In summary, our contributions are:

1. We propose story summarization that requires identifying and extracting multiple plot points from narrative content. This is a challenging multimodal long-video understanding task benefitting from joint analysis of 40+ minute episodes.
2. We pioneer the use of TV show recaps for video understanding and show their application in story summarization. We introduce PlotSnap, a new dataset featuring 2 crime thriller TV series with rich recaps.
3. We propose a novel hierarchical model that features shot and dialog level encoders that feed into an episode-level Transformer. The model operates on the full episode while being lightweight enough to train on consumer GPUs.
4. We present an extensive evaluation: ablation studies validate our design choices, TaleSumm obtains SoTA on PlotSnap and performs well on video summarization benchmarks.

We show generalization across seasons and even across TV shows, and evaluate consistency of labels obtained from multiple diverse sources.

1.2 Summarization Odyssey

Video summarization predates Deep Learning (DL). Past methods focused on generating keyframes [13, 14, 15, 16], skims [17, 18], video storyboards [19], time-lapses [20], montages [21], or video synopses [22]. However, given the effectiveness of DL methods (*e.g*. [23, 24, 25, 26]) over traditional optimization-based approaches, we will primarily discuss learning-based approaches in the following.

Summarization modalities. We classify approaches based on input and output modalities. **(i)** Video to frames/video (V2V) approaches model temporal relations [27, 28, 29], preserve diversity [30, 9], or generate images/videos [31, 32, 33, 34]. On the other, **(ii)** text to text (T2T) methods are either *extractive* [35, 36, 37] picking important sentences from a document, or *abstractive* [38, 39, 40] summarizing the overall meaning by generating new text [41]. Relevant to our work, story screenplay summaries [42, 43] or turning point identification [44] can be seen as T2T summarization.

Multimodal approaches typically benefit from additional modalities to enhance model performance. **(iii)** Video-text to text (VT2T) is popular for screenplays [6, 45], particularly in generating video captions [46, 7, 34]. **(iv)** Video-text to video (VT2V) covers the field of *query-guided summarization* [8, 47, 48]. Finally, the last option is **(v)** video-text to video-text (VT2VT) summarization. Our work lies here and is different from A2Summ [12] as we operate on long videos edited to convey complex stories. Different from trailer generation [49] that avoids spoilers, we wish to identify all key story events.

1.3 Existing Datasets

Summarization datasets. We compare popular summarization datasets based on above modalities in Table 1.1. Video-only datasets, TVSum [2] and SumME [1], consist of short duration videos unlike ours. Other video datasets work with first-person videos [53], are used for title generation [54], and even feature e-sports audience reactions [55]. For a nice overview of text-only (T2T) and text-primary (VT2T) datasets, we recommend reading [6]. Briefly, text datasets include news articles (CNN-DailyMail [51], XSum [52], human dialog (Samsum [56]), and TV/movie screenplays (SummScreen [43]). While similar in spirit to screenplays used for storytelling, PlotSnap is different as it features TV episodes with long videos and dialogs (without speaker labels or scene descriptions), a significant challenge in long-form video understanding.

1.4 Chronicles of Story Summarization: The Untold Saga

Story summarization retrieves multiple sub-stories contained within the story-arc of an episode. To our best knowledge, we do not know of any work that approaches video-text story-summary generation. However, there are approaches to understand stories in movies/TV shows through various subtasks: person identification [57, 58, 59, 60], question-answering [61, 62, 63], captioning [64, 65, 66, 67], situation understanding [68, 69, 70, 71], text alignment [72, 73, 74, 75], or scene detection [76, 77, 78, 79]. Recently, SummScreen3D [6] extends SummScreen [43] with visual inputs, but the output summary is still textual. On the other hand our goal is multimodal story-summary generation by predicting both - important video shots *and* dialogs.

Dataset	Modalities	#	Length	Content	Summary Annotations
SumMe [1]	V2V	25	1-6 min	Holidays, events, sports	Multiple set of key fragments
TVSum [2]	V2V	50	1-11 min	News, how-to, user-generated, documentary	Multiple fragment-level scores
OVP [50]	V2V	50	1-4 min	Documentary, educational, historical, lecture	Multiple set of key-frames
CNN-DailyMail [51]	T2T	311672	766 words	News articles and highlight stories	Human-generated internet summaries
XSum [52]	T2T	226711	431 words	BBC News articles	Single-sentence summary by author
TRIPOD [44]	T2T	99	22072 words	Movies (action, romance, comedy, drama)	Synopses level annotations
SummScreen [43]	T2T	26851	7013 words	TV screenplays (wide scope, 21 genres)	Human-written internet summaries
How2 [7]	VT2T	79114	2-3 min	Instructional videos	Youtube descriptions (and translations)
SummScreen3D [6]	VT2T	4575	5721 words	TV Shows (soap operas)	Human written internet summaries
BLiSS [12]	VT2VT	13303	10.1 min/49 words	Livestream Videos	Human text-summaries; Thumbnail animation
PlotSnap (Ours)	VT2VT	215	40-45 min	TV Shows (crime thriller)	Matching recap shots followed by smoothing

Table 1.1: Overview of video/text/multimodal summarization datasets. # indicates the size of the dataset (no. of instances). The modalities column includes: V2V: Video-to-video, T2T: Text-to-text, VT2T: Video-text to text, and VT2VT: Video-text to video-text summarization. Closest to our work on story summarization are the screenplay datasets (SummScreen, SummScreen3D) that output text summaries.

Chapter 2

PlotSnap: The Plot of a Multimodal Dataset

We introduce the PlotSnap dataset consisting of long-form multimodal TV episodes with a well-structured underlying plot spanning the multiple seasons and episodes. We consider two American crime thriller TV shows with rich storylines: *24* [80] and *Prison Break* (PB) [81]. Unlike sitcoms (used commonly for person id), crime thrillers are recognized for their methodically crafted captivating plot lines. Notably, both *24* and *Prison Break* have good recaps, and are famous for using the catchphrase *Previously on...* before presenting a summary of the previous episode(s).

We present some statistics of PlotSnap in Table 2.1. With a total of 205 episodes, the large number of shots and dialogs present in each episode pose interesting challenges for summarization. The first section of the table presents overall size and duration, second shows statistics for shots and dialog utterances, and the third for recaps. We note that recap shots are much shorter (1.9s *vs*. 3.2s for *24*) allowing the editors to pack more story content in the same duration.

Our key idea is to use professionally edited recaps, shown at the beginning of a new episode, as labels for story summarization. Let $\mathcal{E}_n$ be the n^{th} episode in a TV series. $\mathcal{R}_{n+1}$ is the recap shown just before the episode $\mathcal{E}_{n+1}$ begins and may contain content from all past episodes $\{\mathcal{E}_n, \dots, \mathcal{E}_1\}$. We classify the visual content appearing in the recap into three types: (i) shots that are picked (and usually trimmed) from $\mathcal{E}_n$, (ii) shots that are picked from $\mathcal{E}_{n-1}$ or earlier, and (iii) new shots that did not appear in any previous episode. On average, recap shots from *24* belong to the above three categories with the following proportions: 88%, 5%, and 7%. We remove the last episode of each season due to the absence of a recap.

TV Series	24	Prison Break
# of Seasons	8	2
# of Episodes	172	33
Dataset duration (hours)	125.9	24.0
Avg episode duration (s)	2635 ± 72	2615 ± 39
Avg # of shots per episode	825 ± 101	999 ± 117
Avg duration of shots (s)	3.2 ± 2.5	2.6 ± 2.3
Avg # of utterances per episode	564 ± 54	529 ± 59
Avg # of words/tokens in utterance	7.9 ± 5.4	7.4 ± 5.8
Avg recap duration (s)	104 ± 28	62 ± 20
Avg # of shots in recap	55 ± 12	43 ± 9
Avg # of utterances in recap	33 ± 6	22 ± 5

Table 2.1: Mean ($\pm$ stddev) featuring properties of video shots, dialog utterances, and the recap in our dataset PlotSnap.

2.1 Shot Matching

We propose a novel shot-matching algorithm whose working principle involves frame-level similarities to obtain matches. First, we compute frame-level embeddings using DenseNet169 [82], which was found to work better than models such as ResNet pre-trained on ImageNet [83, 84] based on a qualitative analysis. An example is shown in Fig. 2.1.

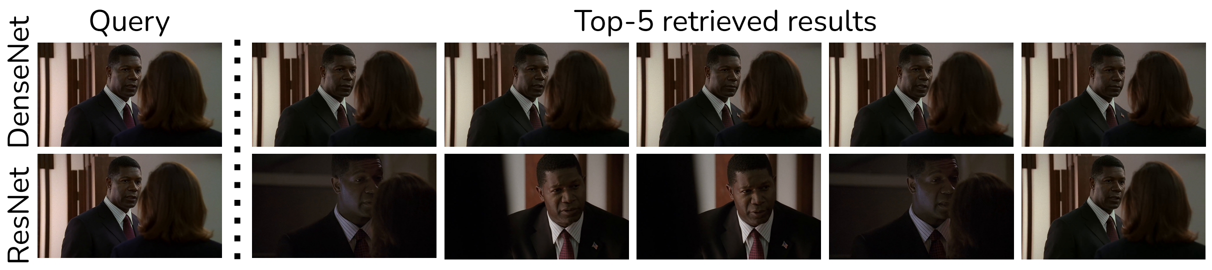

Figure 2.1: Retrieval results for Recap from Episode Frames with DenseNet (Top) v/s ResNet (bottom). We observe qualitatively that DenseNet is able to match to the correct frames from the episode more often.

Second, we discuss how these embeddings are used to obtain matches is detailed below.

Matching. For a given recap shot s in $\mathcal{R}_{n+1}$ we compare it against multiple frames in the episode $\mathcal{E}_n$, and compute a matrix dot-product with appropriate normalization (*cosine similarity*) between respective frame representations of the recap and episode as illustrated in the toy-example of Fig. 2.2. We remove very dark or very bright frames, typical in poor lighting conditions or glares, to avoid spurious matches and noisy labels.

Next, we choose a high threshold to identify matching frames (0.85 in our case after analysis) and fetch all the top matching frames along with their shot indices (the shot where the frame is sourced from). We compute a set union over all matched shots obtained by scoring similarities between the recap frame of shot s and denote this set as $\mathbb{S}_s$. In the example, we match 3 frames of a recap shot and identify several episode shots shown in the blue box with $\mathbb{S}_s$.

Weeding out spurious shot matches. The set $\mathbb{S}_s$ may contain shots beyond a typical shot thread pattern due to spurious matches. These need to be removed to prevent wrong importance scores from being obtained from the recap. To do this, we first find the best matching shot in the episode. We observe that taking top three matched frames for every recap frame results in strong matches. Subsequently, we pick the maximum similarity score for each unique shot matched to a recap frame. This allows us to accumulate the score for an episode shot if multiple frames of the episode shot match with frames of the recap shot. The shot that scores the highest (after summing up the scores) is considered the best-matched episode shot for recap shot s.

Next, we choose a window size of 21 (10 on either side) and include all shots in $\mathbb{S}_s$ that fall within this window to a new matched set, $\mathcal{N}_s$. This is motivated by the typical duration of a scene in a movie or TV episode (40-60 seconds) and an average shot duration of 2-3 seconds. We repeat this process until no more shots are added to the set $\mathcal{N}_s$ and discard the rest in $\mathbb{S}_s$. Thus, for a given shot from the recap, we obtain all matching shots in the episode that are localized to a certain region of high-scoring similarity (see Fig. 2.2). We repeat this process for all frames and shots from the recap.

2.2 Label Smoothing

The intuition behind extending the recap matched shots obtained in the previous section is to include chunks of the sub-story that are important to the storyline. While a short recap (intended to bring back memories) only selects a few shots, a story summary should present the larger sub-story. Selecting only

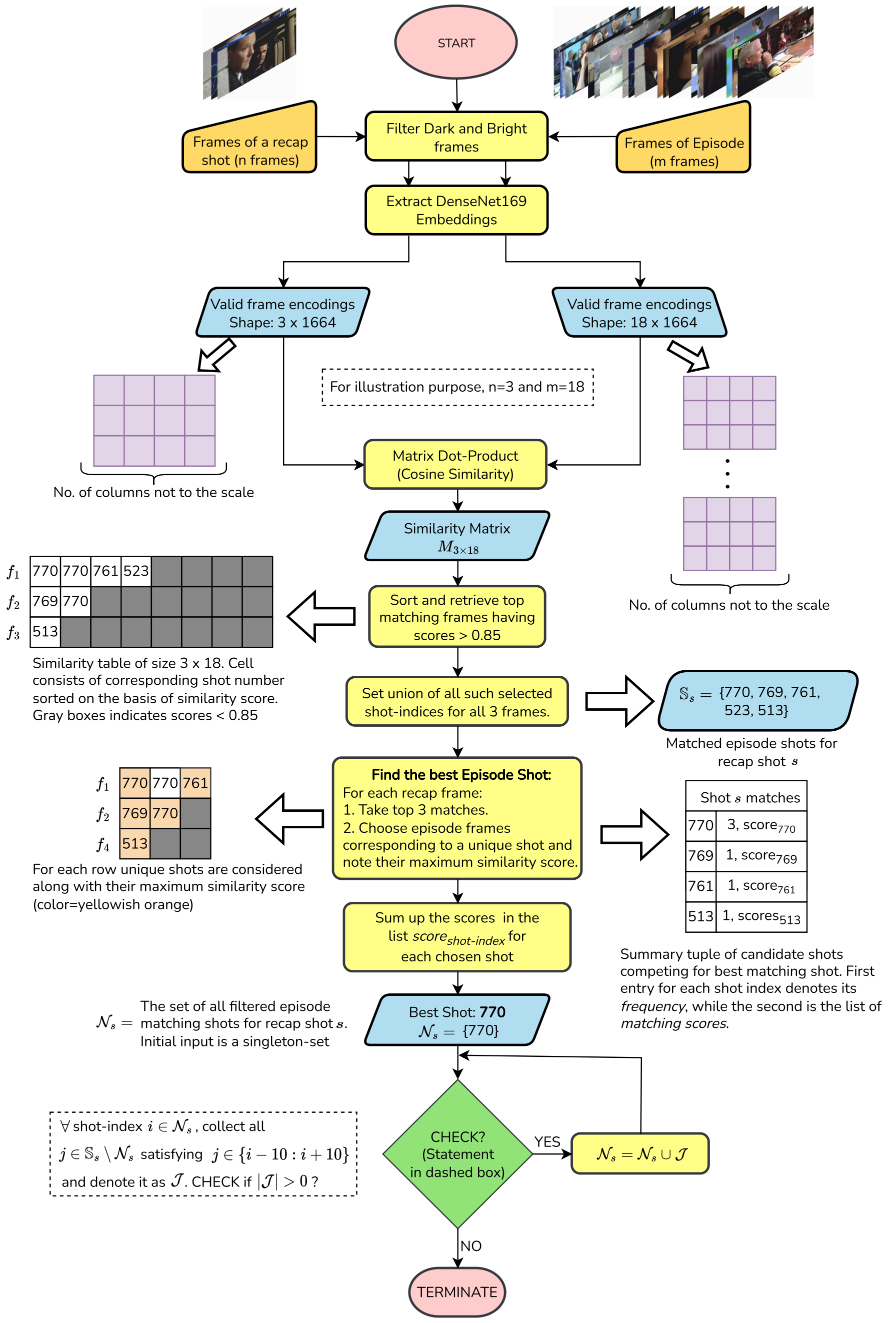
START
Frames of a recap shot (n frames)
Filter Dark and Bright frames
Frames of Episode (m frames)
Extract DenseNet169 Embeddings
Valid frame encodings Shape: 3 x 1664
Valid frame encodings Shape: 18 x 1664
For illustration purpose, n=3 and m=18
No. of columns not to the scale
Matrix Dot-Product (Cosine Similarity)
Similarity Matrix $M_{3\times18}$
No. of columns not to the scale
f_1 770 770 761 523
f_2 769 770
f_3 513
Sort and retrieve top matching frames having scores > 0.85
Similarity table of size 3 x 18. Cell consists of corresponding shot number sorted on the basis of similarity score. Gray boxes indicates scores < 0.85
Set union of all such selected shot-indices for all 3 frames.
$\mathbb{S}_s =$ {770, 769, 761, 523, 513}
Matched episode shots for recap shot s
f_1 770 770 761
f_2 769 770
f_4 513
For each row unique shots are considered along with their maximum similarity score (color=yellowish orange)
Find the best Episode Shot:
For each recap frame:
1. Take top 3 matches.
2. Choose episode frames corresponding to a unique shot and note their maximum similarity score.
Shot s matches
770 3, score$_{770}$
769 1, score$_{769}$
761 1, score$_{761}$
513 1, scores$_{513}$
Sum up the scores in the list $score_{shot\text{-}index}$ for each chosen shot
Summary tuple of candidate shots competing for best matching shot. First entry for each shot index denotes its frequency, while the second is the list of matching scores.
$\mathcal{N}_s =$ The set of all filtered episode matching shots for recap shot s. Initial input is a singleton-set
Best Shot: 770 $\mathcal{N}_s =$ {770}
$\forall$ shot-index $i \in \mathcal{N}_s$, collect all $j \in \mathbb{S}_s \setminus \mathcal{N}_s$ satisfying $j \in \{i-10 : i+10\}$ and denote it as $\mathcal{J}$. CHECK if $|\mathcal{J}| > 0$?
CHECK? (Statement in dashed box)
YES
$\mathcal{N}_s = \mathcal{N}_s \cup \mathcal{J}$
NO
TERMINATE

Figure 2.2: Flowchart for identifying shots from the episode that appear in a recap and can be used as weak labels for story summarization. The process involves identifying the list of high-scoring matching frames, indexing the shots, and then preventing spurious matches by looking for high-scoring matches within a bounded duration. The flowchart presents an example of the process used to identify the set of shots $\mathcal{N}_s$ from the episode that match to the recap shot.

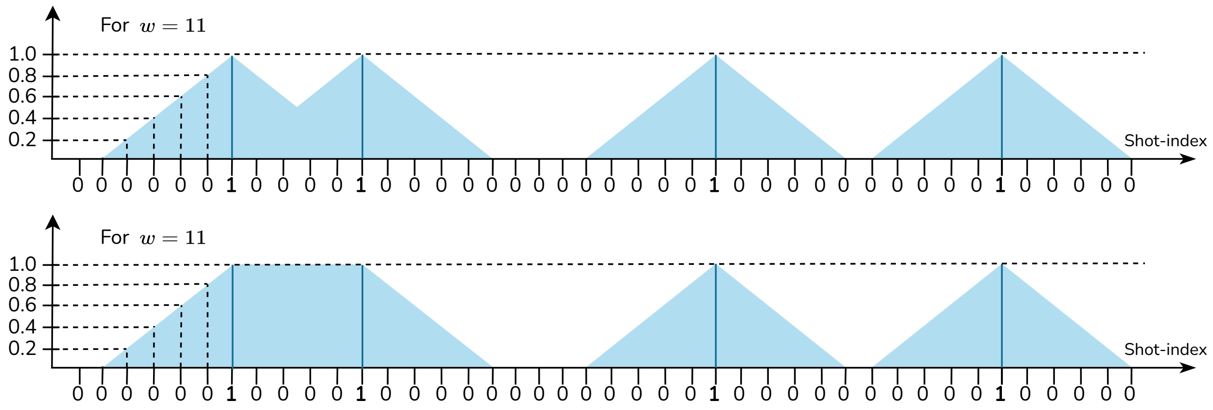

Figure 2.3: Triangle smoothing process. Here *x-axis* denotes binary labels derived from the *shot matching* process, while *y-axis* shows soft *importance scores* used to train our model. **Top:** The triangular filter is applied at each shot selected from the matching process. Scores of shots falling within the window are updated. **Bottom:** In the second step, we add shot importance derived potentially from multiple overlapping triangle filters. This typically happens when episode shots in close proximity are matched to the recap.

one shot in a thread [85] also adversely affects model training due to conflicting signals, as multiple shots with similar appearance can have opposite labels. Label smoothing solves both these issues.

Triangle Smoother. We hypothesize that the importance of shots neighboring a matched shot are usually quite high and use a simple *triangle smoother* to re-label the importance of shots. In particular, we slide a window of size w centering at positive labels and set the importance of neighboring shots according to height of the triangle. The above process is illustrated in Fig. 2.3 as the first step. In the second step, we add and clip the soft labels of strongly overlapping regions to prevent any score from going higher than 1. For the sake of simplicity, we used *triangle* smoothing, however, one could also use other filters. We choose $w = 17$ by analyzing the spread of the shots and their importance scores and comparing them against a few episodes for which we manually annotated the story summaries.

Dialog labels. The above smoothing procedure generates soft labels for video shots. For dialog utterances, we simply import the score of the shot that encompasses the mid-timestamp of the dialog. The key assumption here is that the dialog utterance associated with the matched shot is also important.

2.3 Recap inspired labels.

We present how recap shots and dialogs can be used to create labels for story summarization. First, we manually extract the recap ($\mathcal{R}_{n+1}$) from $\mathcal{E}_{n+1}$ instead of employing automatic detection methods [79] to avoid introducing spurious content. Second, to localize trimmed recap shots in past episodes ($\mathcal{E}_1, \dots, \mathcal{E}_n$), we propose the shot-matching algorithm (see Section 2.1) that conducts pairwise comparisons of frame-level embeddings, making selections based on a threshold determined by similarity score and frequency. Due to shot thread patterns [85], one recap shot may match multiple shots in the episode. This is desirable as we want to highlight larger sub-stories as part of the summary. In fact, selecting only one shot in a thread adversely affects the model due to conflicting signals as shots with similar appearance have different labels.

We think of recap matched shots as temporal point annotations [86]. We identify the set of matching shots in the episode, create a binary label vector, and smooth this vector using a triangular filter. We will refer to these smoothed labels as ground-truth (GT) for story summarization. Extending the supervision helps the model identify meaningful, contiguous sub-stories rather than focusing solely on specific shots highlighted in the recap. For example, it is unlikely that shot s_i is important to the story while $s_{i\pm1}$ is entirely irrelevant (except at scene boundaries). Thus, smoothing is essential to clarify the distinction between positive (essential) and negative (unimportant) shots.

A similar approach can be adopted for dialog utterances. We are able to match 88% of recap utterances to dialog within the smoothed video labels. The rest do not appear in episode $\mathcal{E}_n$ or are picked from extra recorded footage. For simplicity, we inherit labels for the dialogs based on the smoothed label for the temporally co-occurring shot.

Chapter 3

TaleSumm: The Story Crafter

In this chapter, we introduce *TaleSumm*, a sophisticated two-level hierarchical method that feeds upon the whole sequence of video shots and dialog utterances (story elements) in an episode and identifies the important shot *and* utterance. The resulting multi-modal video thus obtained Upon stitching articulate the overall story arc. TaleSumm core architecture is Transformer-based thus allowing flexible hierarchical interaction between video and dialog tokens within local story groups based on a temporal neighborhood (first level) as well as across such temporal groups (second level) for effective message passing.

In the upcoming sections, we will nicely formalize our Problem Statement (Section 3.1), elaborate on Level 1 (Section 3.2) and Level 2 (Section 3.3) of our hierarchical modeling, and conclude it with how-to train and inference (Section 3.4)

Fig. 3.1 shows an overview of the approach we are using in this task.

3.1 Problem Statement

Our aim is to extract a multimodal story summary (video and text) from a given episode, typically lasting around 40 minutes, and encompassing multiple key events.

Notation. An episode $\mathcal{E} = (\mathcal{S}, \mathcal{U})$ consists of a set of N video shots $\mathcal{S} = \{s_i\}_{i=1}^{N}$ and a set of dialog utterances $\mathcal{U} = \{u_l\}_{l=1}^{M}$. A *shot* serves as a basic unit of video processing and comprises temporally contiguous frames taken from the same camera, while a *dialog utterance* typically refers to a sentence uttered by an individual as part of a larger conversation. We denote each shot as $s_i = \{f_{ij}\}_{j=1}^{T_i}$, where f_{ij} are sub-sampled frames, and each utterance as $u_l = \{w_{lp}\}_{p=1}^{T_l}$ with multiple word tokens w_{lp}.

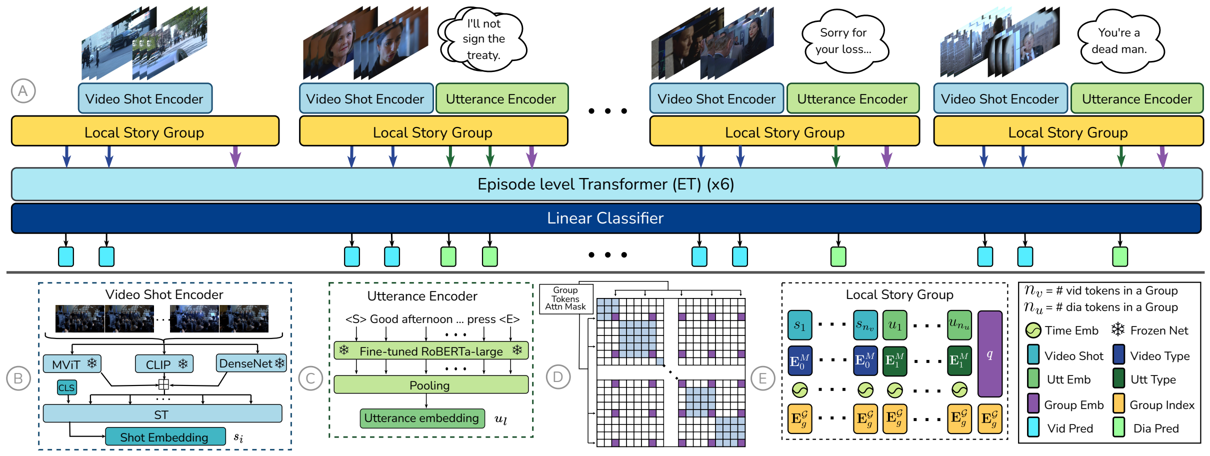

Figure 3.1: **(A) TaleSumm** ingests all video shots and dialogs of the episode and encodes them using (B) and (C). Based on temporal order, we combine tokens into local story groups (*illustration* shows small groups of 2 shots and 0-2 utterances). To each group, we append a group token and add multiple embeddings, before feeding them to the the episode-level Transformer (ET). For each shot or dialog token, a linear classifier predicts its importance. **(B) Video shot encoder.** For each frame, representations from multiple backbones are fused using attention (⊞). We feed these to a shot Transformer encoder ST, and tap a shot-level representation from the CLS token. **(C) Utterance encoder** uses a fine-tuned language model and avg-pooling across all words of the utterance. **(D) Self-attention mask** illustrates the block-diagonal self-attention structure across the episode. Group tokens across the episode (purple squares) communicate with each other. **(E) Multiple embeddings** are added to the tokens to capture modality type, time, and membership to a local story group.

Summarization as importance scoring. While humans may naturally select start and end temporal boundaries to indicate important sub-stories, for ease of computation, we discretize time and associate an importance score with each video shot or dialog utterance. Thus, given an episode $\mathcal{E} = (\mathcal{S},\mathcal{U})$, we formulate story summarization as a binary classification task applied to each element (shot or dialog). The ground-truth labels can be denoted as $\mathbf{y}^{\mathcal{S}} = \{y_i^{\mathcal{S}}\}_{i=1}^{N}$ and $\mathbf{y}^{\mathcal{U}} = \{y_l^{\mathcal{U}}\}_{l=1}^{M}$, where each $y_i^{\mathcal{S}}, y_l^{\mathcal{U}} \in [0, 1]$, signaling their importance to the story summary. While the ground-truth annotations may be indicated through a temporal start-end range, we convert them into discrete labels at a shot-level and dialog utterance-level granularity to convert raw continuous time-space into discrete ground-truth labels.

3.2 Level 1: Shot and Dialog Representations

In narrative video production, *shots* play an important role in advancing the storyline and contextualizing neighboring content. We obtain shot-level representations from granular frame-level features to determine how well the shot can contribute to understanding the storyline.

Feature extraction. To capture various aspects of the shot, we use *three* pretrained backbones that capture visual diversity through people, their actions, objects, places, and scenes: $\phi^k_{\mathcal{S}}(\cdot), k = \{1, 2, 3\}$. We extract relevant visual information from frame(s) of a given shot, s_i as follows:

$$\mathbf{f}^k_{ij} = \phi^k_{\mathcal{S}}\left(\{f_{ij}\}\right) , \quad \mathbf{f}^k_{ij} \in \mathbb{R}^{D^k_{\mathcal{S}}} . \tag{3.1}$$

Note that the backbone may encode a single frame f_{ij} or a short sequence around f_{ij}.

For dialog utterances, we adopt a fine-tuned language model $\phi^{\text{FT}}_{\mathcal{U}}$, to compute contextual word-level features:

$$\mathbf{w}_{lp} = \phi^{\text{FT}}_{\mathcal{U}}\left(\{w_{lp}\}\right) , \quad \mathbf{w}_{lp} \in \mathbb{R}^{D_{\mathcal{U}}} . \tag{3.2}$$

Shot CLS **pooling.** To compute an aggregated shot representation, we combine frame-level signals into a compact representation. An attention-based aggregation ($\boxplus$) (inspired by [87]), effectively weighs the most pertinent information (*e.g*. action in a motion-heavy shot or scenery in an establishing shot). First, the frame features from different backbones are projected to the same space (using $\mathbf{W}^k_{\mathcal{S}} \in \mathbb{R}^{D \times D^k_{\mathcal{S}}}$) and then concatenated to form $\hat{\mathbf{f}}^{1:3}_{ij} \in \mathbb{R}^{3D}$ (Eq. 3.3). A linear layer $\mathbf{W}_{\mathcal{P}} \in \mathbb{R}^{3 \times 3D}$ followed by $\tanh$ and $\mathsf{softmax}$ computes scalar importance scores that are used for weighted fusion:

$$\hat{\mathbf{f}}^{1:3}_{ij} = [W^1_{\mathcal{S}}\mathbf{f}^1_{ij}, W^2_{\mathcal{S}}\mathbf{f}^2_{ij}, W^3_{\mathcal{S}}\mathbf{f}^3_{ij}] , \tag{3.3}$$

$$\boldsymbol{\alpha}^{1:3}_{ij} = \mathsf{softmax}(\tanh(W_{\mathcal{P}}\hat{\mathbf{f}}^{1:3}_{ij})) , \tag{3.4}$$

$$\mathbf{F}_{ij} = \alpha^1_{ij}\hat{\mathbf{f}}^1_{ij} + \alpha^2_{ij}\hat{\mathbf{f}}^2_{ij} + \alpha^3_{ij}\hat{\mathbf{f}}^3_{ij} . \tag{3.5}$$

We omit bias for brevity. We add relative frame position to $\mathbf{F}_{ij}$ through a *time-embedding* vector, $\mathbf{E}^{\mathcal{S}}_j$, similar to Fourier position encoding [88].

A shot transformer [88] ST is used to encode the frame-level feature sequence $\{\mathbf{F}_{ij}\}^{T_i}_{j=1}$. We tap the output from the CLS token appended at the beginning of the sequence (*e.g*. similar to BERT [89]) as the final shot representation:

$$\mathbf{s}_i = \mathsf{ST}(\{\mathbf{F}_{ij} + \mathbf{E}^{\mathcal{S}}_j\}^{T_i}_{j=1}) , \quad \mathbf{s}_i \in \mathbb{R}^D . \tag{3.6}$$

Dialog utterance representation. First, we project the tokens $\mathbf{w}_{lp}$ to $\mathbb{R}^D$ using a linear layer $\mathbf{W}_\mathcal{U} \in \mathbb{R}^{D\times D_\mathcal{U}}$. As the tokens are already contextualized by $\phi_\mathcal{U}^{\text{FT}}$, a simple mean-pool across the p tokens is found to work well:

$$\mathbf{u}_l = \text{mean}_p(\{\mathbf{W}_\mathcal{U}\mathbf{w}_{lp}\}_{p=1}^{T_l}), \quad \mathbf{u}_l \in \mathbb{R}^D . \tag{3.7}$$

3.3 Level 2: Episode-level Interactions

We propose an episode-level Transformer encoder, ET, that models interactions across shots and dialog of the entire episode. Predicting the importance of an element (shot or dialog) requires context in a neighborhood; *e.g*. shot in a scene, dialog utterance in a conversation.

Additional embeddings. We arrange shot and dialog tokens based on their order in the episode (see Fig. 3.1(A)). Learnable *type embeddings* help the model distinguish between shot and dialog modalities ($\mathbf{E}^M \in \mathbb{R}^{2\times D}$).

We encode the real time (in seconds) of appearance of each element (shot or dialog) using a binning strategy. Given an episode of T seconds, we initialize Fourier position encodings $\mathbf{E}^T \in \mathbb{R}^{\lceil T/\tau\rceil\times D}$ where τ is the bin-size (hyperparameter). Based on the mid-timestamp of each element t, we add $\mathbf{E}^T_t$ to the representation, the $\lfloor t/\tau\rfloor^{\text{th}}$ row in the position encoding matrix. Such time embeddings allow our model to: (i) implicitly encode shot duration; and (ii) relate co-occurring dialogs with video shots without the need for complex attention maps.

Local story groups. The total number of video shots and dialog make up the sequence length, $S{=}N{+}M$, can be around 1500 tokens for ET. Self-attention over so many tokens is not only computationally demanding, but can also be difficult to train due to noisy and unrelated distant tokens. We adopt a block-diagonal attention mask to constrain the tokens to attend to local story regions:

$$\mathbf{A}_{S\times S} = \text{diag}(\mathbb{1}_{n_1\times n_1}, \ldots, \mathbb{1}_{n_g\times n_g}, \ldots, \mathbb{1}_{n_G\times n_G}), \tag{3.8}$$

where $\mathbb{1}_{n_g\times n_g}$ denotes an all one matrix, n_g is the # of tokens in the g^{th} local block, $\sum_{g=1}^{G} n_g = S$, and diag$(\ldots)$ constructs a block diagonal matrix. Further, we add new learnable group index embeddings $\mathbf{E}^\mathcal{G} \in \mathbb{R}^{G\times D}$ to our tokens to inform our model about their group membership.

Group tokens. While capturing interactions across all tokens may lead to poor performance, self-attention only within the local story groups prohibits the model from capturing long-range story dependencies. To enable story group interactions, we propose to add a set of *group tokens* to the input, extending the sequence length to $\hat{S}{=}S{+}G$. The group tokens $\mathbf{q}_g$ represent an additional layer of hierarchy within the episode model as they summarize the story content inside a group and also communicate across groups, providing a way to understand the continuity of the story. Fig. 3.1(E) shows how group tokens are inserted at the end of each local story group's shot and dialog tokens.

To facilitate cross-group communication, we make two modifications to the self-attention mask: (i) The size of each local group n_g is extended by 1 to incorporate the group token $\mathbf{q}_g$ within the block matrix. We assume $\mathbf{A}$ is updated to reflect this and is of size $\hat{S} \times \hat{S}$. (ii) We compute a binary index $\mathbf{o} \in \{0,1\}^{\hat{S}}$ to represent the locations at which a group token appears in the sequence. The new self-attention mask $\hat{\mathbf{A}} = \mathbf{A}{+}\mathbf{o}\mathbf{o}^T$ allows group-tokens to communicate. Fig. 3.1(D) illustrates the attention mask; light blue squares correspond to attention within a group, and sparse purple squares visualize the group tokens.

Importance prediction. We present how shot or dialog scores can be estimated. First, the input tokens to ET are:

$$\begin{aligned} \hat{\mathbf{s}}_i &= \mathbf{s}_i + \mathbf{E}^M_0 + \mathbf{E}^T_{t_i} + \mathbf{E}^{\mathcal{G}}_{g_i}\,, && (3.9)\\ \hat{\mathbf{u}}_l &= \mathbf{u}_l + \mathbf{E}^M_1 + \mathbf{E}^T_{t_l} + \mathbf{E}^{\mathcal{G}}_{g_i}\,, && (3.10)\\ \mathbf{q}_g &= \mathbf{q} + \mathbf{E}^{\mathcal{G}}_g\,. && (3.11) \end{aligned}$$

where t_i, t_l and g_i, g_l correspond to the mid-timestamp and group membership of shot s_i and dialog u_l respectively. $\mathbf{q}$ denotes the learnable shared group type embedding.

We feed the updated shot, dialog, and group token representations to ET post LayerNorm [90], a H_{E} layer Transformer encoder with a curated self-attention mask $\hat{\mathbf{A}}$:

$$[\ldots, \tilde{\mathbf{s}}_i, \tilde{\mathbf{u}}_l, \tilde{\mathbf{q}}_g, \ldots] = \mathsf{ET}([\ldots, \hat{\mathbf{s}}_i, \hat{\mathbf{u}}_l, \mathbf{q}_g, \ldots]; \hat{\mathbf{A}})\,, \quad (3.12)$$

with all tokens, *i.e.* $\{i\}_1^N$, $\{l\}_1^M$, and $\{g\}_1^G$.

After ET, we compute shot and dialog importance scores using a linear classifier $\mathbf{W}^C \in \mathbb{R}^{1\times D}$ followed by sigmoid function $\sigma(\cdot)$:

$$\hat{y}^{\mathcal{S}}_i = \sigma(\mathbf{W}^C \tilde{\mathbf{s}}_i) \text{ and } \hat{y}^{\mathcal{U}}_l = \sigma(\mathbf{W}^C \tilde{\mathbf{u}}_l)\,\forall i, l\,. \quad (3.13)$$

3.4 Training and Inference

Training. TaleSumm is trained in an end-to-end fashion with *BinaryCrossEntropy* (BCE) loss. We provide positive weights, w (ratio of negatives to positives) to account for class imbalance. Modality specific losses are added:

$$\mathcal{L} = \mathsf{BCE}\left(\hat{\mathbf{y}}^{\mathcal{S}}, \mathbf{y}^{\mathcal{S}}; w^{\mathcal{S}}\right) + \mathsf{BCE}\left(\hat{\mathbf{y}}^{\mathcal{U}}, \mathbf{y}^{\mathcal{U}}; w^{\mathcal{U}}\right) \ . \tag{3.14}$$

Inference. At test time, we follow the procedure outlined in Section 3.3 and generate importance scores for each video shot and dialog utterance (Eq. 3.13).

Variations. As we will see empirically, our model is versatile and well-suited for adding/removing modalities or additional representations by adjusting the sequence length of the Transformer (number of tokens). It can also be modified to act as an unimodal model that applies only to video or dialog utterances by disregarding other modalities.

Chapter 4

Experiments and Discussion

In this section, we will start with our experimental setup in Section 4.1 that elucidates which dataset splits and evaluation metric we used, followed by implementation details in Section 4.2. Subsequently, we conduct a series of ablation studies which motivated and finalized the design choices for our model while we compare against various adapted SoTA models in unimodal and multimodal setting 4.3. Ultimately, we conclude this chapter with some qualitative analysis and discussion in Section 4.4. Here we shed some light on our model performance on benchmark datasets, its generalization ability to other TV Shows, and the quality of our recap-inspired labels.

4.1 Setup

Data splits. We present 4 split-settings as elaborated below for evaluating its generalizability and robustness.

1. IntraCVT: On *24*, most experiments follow an intra-season 5-fold cross-validation-test strategy. It represents 5 different non-overlapping splits from 7 seasons of *24* (Season 2 to 8). *Intra-season* means episodes from each season are present in the train/val/test splits. For example, split-1 uses 5 episodes from the end of each season for val and test (2, 3 respectively). Likewise, Split-5 (the fifth fold) uses episodes from the beginning of the season in val/test. We observe that Split-5 is harder.
2. X-Season: On *24*, we assess cross-season generalization through a 7-fold cross-validation-test. involves 7 non-overlapping splits with one season entirely kept for testing (from Season 2 to 8), while the train and val use 18 and 5 episodes respectively from each season. This strategy is used to test for the generalization of our model on different seasons of the same series, *24*.

3. X-Series: We show transfer results from *24* to *Prison Break* to assess the generalizability of the model in different settings where plotlines from disparate seasons or even separate series may diverge significantly due to distinct directors and producers, with diverse casting roles and unique methods of recap construction. It includes 8 seasons (seasons 2 to 9) from *24* in train/val (19/4) and 2 seasons (seasons 2 and 3) from *PB* for the test. This split is designed to check the effectiveness of our model across TV series.
4. *default-split*: consists of a single non-overlapping train/val/test split with 126/18/17 episodes from *24*, respectively. This split is used for comparison against labels from *Fandom* or *Human* annotations.

Evaluation metric. We adopt Average Precision (AP, area under precision-recall (PR) curve) as the metric to compare predicted importance scores of shots or dialogs against ground-truth.

4.2 Implementation details

Feature representations serve as the key ingredient for providing the desired signals to our model so as to comprehend the intricate blend of multi-modal signals and effectively decipher the genuine plotline through extraction and analysis. In the following, we describe different backbones used to extract features for video shots and dialog utterances.

Visual features. Prior to the extraction of frame(s)-level features, we first segment the episode into shots using DFD [91] (Displaced Frame Difference) algorithm. We adopt 3 specialized backbones, that combinedly perform best (as shown in feature combination ablations in Table 4.2): (i) DenseNet169 [82] pre-trained on ImageNet [83], SVHN [92], and CIFAR [93] for object semantics; (ii) MViT [94] pre-trained on Kinetics-400 [95] for action information; and (iii) OpenAI CLIP [96], pre-trained on 4M image-text pairs, for semantic information.

Utterance features. We adapt RoBERTa-large [97] for extractive summarization using parameter-efficient fine-tuning [5, 6] on the text from our dataset. Given dialogs from the episode, our fine-tuning objective is to predict the important dialogs. We extract word/token-level representations ($\mathbf{w}$) from *finetuned* (but frozen) RoBERTa-large ($\phi_{\mathcal{U}}^{\text{FT}}$) for the task of dialog story summarization. *Notably*, the absence of speaker annotations in our use of raw dialog texts poses a challenge in determining the speaker of each utterance and discerning how the storyline is evolving.

Feature Extraction Here we present additional details of the backbones used for feature extraction for video as well as dialog modalities. Prior to feature extraction, setting an appropriate fps for every video is important to trade off between capturing all aspects of the video while keeping computational load low. We find 8 fps to be a good balance between the two.

Visual Feature Backbones. We present the details for three backbones capturing different aspects of a video.

DenseNet169 $\mathbf{f}^1$. Feature extraction of salient objects/person in each frame is of utmost priority and is achieved through DenseNet169 [82] pretrained on ImageNet [83], SVHN [92], and CIFAR [93]. We consider the frozen backbone without the linear classification head to obtain flattened features, $\mathbf{f}^1 \in \mathbb{R}^{1664}$. Before feeding the images, we apply a few preprocessing steps to sub-sample raw images.

1. Frames are resized to 256×256 resolution along with center cropping.
2. RGB pixel values are scaled to $[0, 1]$ followed by mean and standard deviation normalization.

We use the architecture as well as parameters from PyTorch Hub[1], version `pytorch/vision:v0.10.0`.

MVIT $\mathbf{f}^2$. Beyond objects/person, their actions too affect the importance of a shot, and hence having them serves the purpose of representing a shot from a different perspective. For this, we use MViT [94] pretrained on Kinetics-400 [95]. We feed the original video with some pre-processing as explained above to obtain feature embeddings, $\mathbf{f}^2 \in \mathbb{R}^{768}$. With a window-size=32 and stride=16, we extracted per-window encodings while padding zeros at the end (black-frames) to account for selecting window-size amount of frames.

1. Frame resizing to 256×256 followed by center-cropping to 224×224 resolution.
2. Pixel scaling from *8-bit* format to *float* format ($[0, 1]$). Following this, mean and standard deviation normalization is performed.
3. We chose MViT-Base 32×3 that ingests a chunk of video (32 frames) at once and produces an embedding vector.

We import the architecture and pretrained parameters from PytorchVideo[2].

CLIP $\mathbf{f}^3$. This is a multi-modal backbone that can produce representations corresponding to matching textual descriptions. We borrow the CLIP [96] pre-trained model from the huggingface [98] library and use their inbuilt image processor as well as feature extractor to obtain subsampled frame-level encodings, $\mathbf{f}^3 \in \mathbb{R}^{512}$.

[1]`https://pytorch.org/hub/`
[2]`https://pytorchvideo.org/`

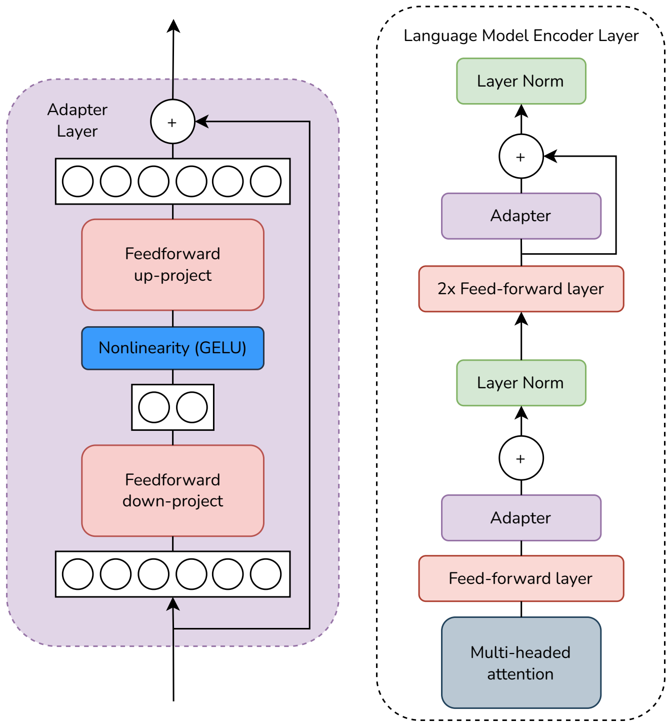

Figure 4.1: Architecture of the adapter module [5] and its integration with language model's encoder layer. **Left**: The adapter module has fewer parameters compared to the attention and feedforward layers in a Transformer layer and consists of a bottleneck and skip-connection. **Right**: We add the adapter module twice to each encoder layer. Once after the multi-head attention and the other is placed after the two feed-forward layers typical of a Transformer architecture. For the Transformer *decoder*, in the case of PEGASUS$_{\text{LARGE}}$, we add one extra adapter after cross-attention as well. **Adaptation**: When adapting the model, the purple and green layers illustrated on the right module are trained on the downstream data and task, while the other blocks are frozen.

1. Short-side is resized to 224 pixels followed by center-cropping (to 224$\times$224).
2. Re-scaling 8-bit image to $[0, 1]$ interval.
3. Mean and standard deviation normalization.

Dialog Features We test three different dialog features and present the details as follows.

Fine-tuning Language Models. We fine-tune language models such as PEGASUS$_{\text{LARGE}}$ [40], RoBERTa-Large [97], and MPNet-base-v2 [99] for our task of extractive dialog summarization. To account for the small dataset sizes, for all fine-tuning, we use the Adapter modules [5] that add only a few trainable parameters in the form of down- and up-projection layers as illustrated in Fig. 4.1.

RoBERTa-Large [97] and MPNet-base-v2 [99] are fine-tuned for utterance-level binary classification to decide whether the dialog utterance is important or not.

PEGASUS$_{\text{LARGE}}$ is trained originally to generate abstractive summaries. Instead, we adapt it to generate summary dialogs. As the number of tokens accepted by the PEGASUS$_{\text{LARGE}}$ model does not allow feeding all the dialog from the episode, we break it into 6 chunks.

Word-level embeddings. We use *last hidden-state* of the encoder to obtain contextualized word-level embeddings, $\mathbf{w} \in \mathbb{R}^{1024}$ (768 for MPNet-base-v2). PEGASUS$_{\text{LARGE}}$, being a generative model (consisting of both encoder and decoder), we keep only the encoder portion for word-level feature extraction.

Frame sampling strategy. We *randomly* sample up to 25 frames per shot during training as a form of data augmentation. This confers an advantage in terms of augmentation that enrich our shot encodings. Relative indices of these frames in the given shot serve as positional embeddings to the respective frame features. During inference, we use *uniform* sampling. Fourier position embeddings $\mathbf{E}_j^{\mathcal{S}}$ use the video frame index.

Architecture details. We experiment with the number of layers for ST, $H_{\mathsf{S}} \in [1:3]$ and ET, $H_{\mathsf{E}} \in [1:9]$, and find $H_{\mathsf{S}}=1$ and $H_{\mathsf{E}}=6$ to work best. Except the number of layers, ST and ET have the same configuration: 8 attention heads and $D=128$. Appropriate padding and masking is used to create batches. We compare multiple local story group sizes $n_g \in \{5:30:5\}$ and find $n_g=20$ to work best.

Training details. Our model is implemented in PyTorch [100] and trained on 4 RTX-2080 Ti GPUs for a maximum of 65 epochs with a batch size of 4 (*i.e.* 4 entire episodes). We adopt the AdamW optimizer [4] with parameters: learning rate$=10^{-4}$, weight decay$=10^{-3}$. We use OneCycleLR [101] as learning rate scheduler with max lr$=10^{-3}$, and multiple dropouts [102]: 0.1 for projection to 128 dim inside video and utterance encoder; 0.2 for attention layers; and 0.2 for the classification head. The hyperparameters are tuned for best performance on validation set.

4.3 Experiments on *24*

4.3.1 Architecture ablations.

Results in Table 4.1 are across two dimensions: (i) columns span the shot or utterance level and (ii) rows span the episode level. All model variants outperform a baseline that predicts a random score between $[0, 1]$ – 34.2 (video) and 30.4 (dialog), averaged over 1000 trials.

	Video-only					Dialog-only		
	Avg	Max	Cat	Tok	⊞	Max	Avg	wCLS
MLP	42.3	42.3	42.3	42.4	42.5	35.7	35.7	35.8
woG + FA	51.8	51.9	51.1	51.6	52.0	44.5	44.5	44.6
wG + SA	52.4	52.5	53.3	53.3	**53.4**	46.5	*46.5*	**47.2**

Table 4.1: **Rows** demonstrate methods for capturing episode-level interactions. In an MLP, tokens are independent. woG+FA is a Transformer encoder that captures full-attention over the entire episode without grouping; and wG+SA uses the proposed architecture with local story groups and sparse-attention. **Columns** describe the *aggregation* method used to combine frame (or token) level features into shot (or utterance) representation. C1 and C7 use average pooling. C2 and C6 use max pooling. C3-C5 are variants of ST: C3 concatenates backbone features of each frame, C4 uses backbone features as separate tokens, and C5 uses proposed ⊞ attention fusion. C8 uses the CLS token for dialog. Chosen: ⊞ for shot, and average pooling for utterance representation.

Across rows, we observe that the MLP performs worse than the other two variants by almost 10% AP score because assuming independence between story tokens is bad. Our proposed approach with local story groups and sparse-attention (wG+SA) outperforms a vanilla encoder without groups and full-attention (woG+FA) by 1-2% on the video model and 2-3% for the dialog model.

Across columns, performance changes are minor. However, when using wG+SA at the episode-level, gated attention fusion with a shot transformer (⊞) improves results over Avg and Max pooling by 1%. For dialog-only, though wCLS outperforms Avg and Max by 0.7%, we adopt Avg pooling for its effective performance in a multimodal setup.

Extended Feature and Architecture Ablations. Here we provide extensive ablation studies with regard to feature combinations and model hyperparameters. All experiments are run on *IntraCVT* split, and we report mean and standard deviation.

Visual features for Story summarization. Table 4.2 shows the results for all combinations of the three chosen visual feature backbones in a multimodal setup. We draw two main conclusions: (i) The Shot Transformer encoder (ST) shows improvements when compared to simple average or max pooling. (ii) DMC (DenseNet + MViT + CLIP), the feature combination that uses all backbones, performs well,

Pooling		Avg	Max	Cat	Tok	Stack	$\boxplus$
Concatenate		✓	✓	✓	NC	✓	✓
DMC	Video AP	54.1 ± 4.5	54.1 ± 4.6	*54.2* ± 4.1	54.1 ± 4.0	53.9 ± 4.7	**54.2** ± 3.3
	Dialog AP	48.7 ± 4.1	48.8 ± 4.4	48.9 ± 4.7	*49.0* ± 4.8	*49.1* ± 4.8	**49.0** ± 4.9
DM	Video AP	54.0 ± 3.8	54.1 ± 3.3	54.0 ± 4.1	53.9 ± 3.2	53.6 ± 4.3	53.7 ± 3.0
	Dialog AP	48.6 ± 4.4	48.8 ± 4.4	48.9 ± 4.4	48.6 ± 3.6	48.4 ± 4.8	48.6 ± 4.3
MC	Video AP	53.8 ± 3.6	53.9 ± 4.0	53.8 ± 4.5	54.0 ± 4.9	53.8 ± 4.2	**54.2** ± 4.3
	Dialog AP	48.5 ± 4.6	48.8 ± 4.1	48.7 ± 4.5	*49.0* ± 4.1	48.5 ± 4.7	**49.1** ± 4.2
DC	Video AP	54.0 ± 4.5	54.1 ± 4.3	54.1 ± 4.0	54.0 ± 4.0	54.1 ± 3.9	54.1 ± 4.2
	Dialog AP	48.2 ± 4.9	48.7 ± 4.5	48.9 ± 4.7	*49.0* ± 4.6	*49.0* ± 4.8	*49.0* ± 4.9
D	Video AP	53.5 ± 4.2	53.4 ± 4.2	54.0 ± 3.8	-	-	-
	Dialog AP	48.1 ± 4.2	48.6 ± 4.3	48.9 ± 4.4	-	-	-
M	Video AP	52.9 ± 3.2	53.6 ± 3.6	53.5 ± 3.4	-	-	-
	Dialog AP	48.2 ± 4.0	48.4 ± 3.9	48.7 ± 4.6	-	-	-
C	Video AP	53.9 ± 3.7	53.9 ± 3.9	54.1 ± 3.7	-	-	-
	Dialog AP	48.6 ± 4.2	48.7 ± 4.2	48.7 ± 4.0	-	-	-

Table 4.2: TaleSumm ablations for different feature combination strategies, using **both video and dialog modalities**. Feature ablations are performed over the visual modality. Columns describe the *Pooling* approaches used to form shot-level from frame-level representations; *Concatenate* corresponds to how backbone feature are combined (concatenate ✓ or as separate tokens (NC)). *Stack* pooling is an alternative approach to $\boxplus$, where instead of condensing the aggregated (concatenated) visual features via a linear layer $\mathbf{W}_{\mathcal{P}} \in \mathbb{R}^{3\times 3D}$, we obtain individual feature importance score (with $\mathbf{W}_{\mathcal{P}} \in \mathbb{R}^{1\times D}$ followed by $\tanh$ and $\mathsf{softmax}$). **D**: DenseNet169, **M**: MViT, and **C**: CLIP. All results are for TaleSumm that captures *Episode* level interactions.

while MC shows on-par performance. Importantly, the feature combination is better than using any feature alone.

Language backbones for Story summarization. Different from the previous experiment, Table 4.3 shows results for different dialog features with different word-to-utterance pooling approaches in a multimodal setting. RoBERTa-Large outperforms all other language models. Nevertheless, the other models are not too far behind.

Pooling		Max	Avg	wCLS
PEGASUS$_{\text{LARGE}}$ [40]	Video AP	52.7 $\pm$ 4.3	53.1 $\pm$ 4.2	53.1 $\pm$ 4.7
	Dialog AP	47.9 $\pm$ 3.6	48.0 $\pm$ 5.1	47.9 $\pm$ 4.9
MPNet-Base [99]	Video AP	53.2 $\pm$ 3.9	53.1 $\pm$ 3.7	53.5 $\pm$ 3.4
	Dialog AP	48.0 $\pm$ 4.4	47.2 $\pm$ 3.2	48.6 $\pm$ 5.0
RoBERTa-Large [97]	Video AP	54.1 $\pm$ 3.8	**54.2** $\pm$ 3.3	54.1 $\pm$ 4.2
	Dialog AP	*49.0* $\pm$ 4.6	**49.0** $\pm$ 4.9	*49.0* $\pm$ 4.3

Table 4.3: TaleSumm ablations for various dialog utterance feature backbones. Visual features are fixed to DMC.

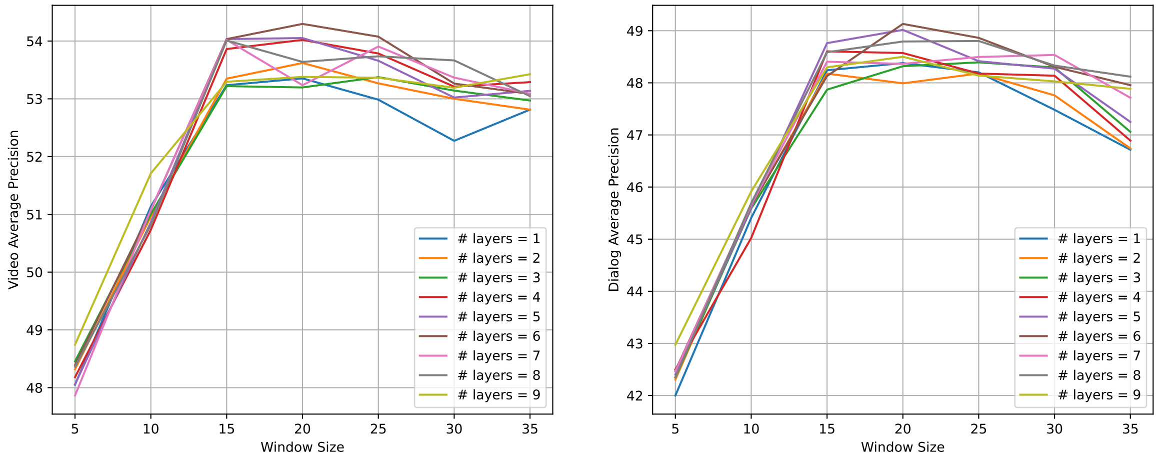

Figure 4.2: Performance of TaleSumm for varying local story group size (also referred to as window size) and number of layers in the episode-level Transformer ET. y-axis denotes the AP score for the **Left**: Video and **Right**: Dialog. We observe that a 6 layer model $H_{\text{E}} = 6$ works well together with a local story group size of $n_g = 20$.

Number of ET layers and local story group size. Two important hyperparameters for our model are the number of layers H_{E} in the ET and the local story group size n_g. Fig. 4.2 shows the performance on video AP (left) and dialog AP (right) with a clear indication that 6 layers and a story group size of 20 shots are appropriate.

	Model	AttnMask	GToken	Video AP	Dialog AP
1	Video-only	SA	✓	53.4 ± 3.9	-
2	Dialog-only	SA	✓	-	47.2 ± 3.9
3	TaleSumm	FA	-	51.8 ± 3.6	43.8 ± 4.7
4		FA	✓	51.9 ± 3.7	44.0 ± 4.6
5		SA	-	53.9 ± 3.4	48.8 ± 4.6
6		SA	✓	**54.2** ± 3.3	**49.0** ± 4.9

Table 4.4: TaleSumm ablations. The *AttnMask* column indicates if the self-attention mask is applied over the full episode (FA) or uses sparse block diagonal structure of story groups (SA). The *GToken* indicates whether the group token is absent (-) or present (✓). R6 is our final chosen model for subsequent experiments.

4.3.2 TaleSumm ablations

The results are presented in Table 4.4. Rows 1 and 2 highlight the best video-only and dialog-only models (from Table 4.1). We report mean ± std dev on the val set. Standard deviation is found to be high due to variation across multiple folds; but low across random seeds. Results for joint prediction of video shot and utterance importance are shown in rows 3-6. Our proposed approach in row 6 performs best for both modalities, outperforming rows 3-5.

4.3.3 SoTA comparison.

We compare against SoTA methods: video-only (PGLSUM [103], MSVA [104]), dialog-only (PreSumm [35]), and multimodal (A2Summ [12]) in Table 4.5. While none of the above methods are built for processing 40 minutes of video, we make small modifications to them to make them comparable to our work (details in Section 4.3.4). On both the validation and the test set, TaleSumm outperforms all other baselines in both modalities.

PreSumm [35] extends BERT by considering multiple sentences and [CLS] tokens at once, allowing contextualization across them and tapping out those [CLS]s for predicting importance score. Although there are architectural similarities with ET, the need for complementary modality poses a challenge. The performance drop observed in A2Summ [12] is attributed to using a dual-contrastive loss and long sequence samples, which inherently add noise. Aligning episodic visual and text content while contrasting

Model	Val		Test	
	Video AP	Dialog AP	Video AP	Dialog AP
PGLSUM [103]	48.8 ± 3.3	-	47.1 ± 2.4	-
MSVA [104]	47.3 ± 3.8	-	45.5 ± 1.2	-
Video-Only	53.4 ± 3.9	-	**50.6** ± 3.6	-
PreSumm [35]	-	43.1 ± 3.3	-	41.6 ± 2.0
Dialog-Only	-	47.2 ± 3.9	-	43.4 ± 2.8
A2Summ [12]	35.1 ± 1.8	33.2 ± 2.8	33.8 ± 1.7	31.6 ± 2.2
TaleSumm (Ours)	**54.2** ± 3.3	**49.0** ± 4.9	*50.1* ± 2.8	**46.0** ± 2.1

Table 4.5: Comparison against SoTA video-only, text-only, and multimodal summarization models. Our approach outperforms previous work by a significant margin.

it with other episodes mess up the representations. Nevertheless, TaleSumm enables multimodal story summarization, allowing the generation of video-text labels (VT2VT), unlike previous approaches.

4.3.4 Adapting SoTA Approaches for PlotSnap

In this section, we discuss how we adapt different video- and dialog-only state-of-the-art baselines such as MSVA [104], PGL-SUM [103], PreSumm [35], and A2Summ [12] for comparison with TaleSumm.

MSVA [104] considers frame-level features from multiple sources and applies aperture-guided attention across all such feature streams independently, followed by intermediate fusion and a linear classification head that selects frames based on predicted importance scores. Since we are modeling at the level of the entire episode, we feed condensed shot features (after Avg or Max pooling or ST) of each backbone: DenseNet169 (X_o), MViT (X_r), and CLIP (X_f) through *three* different input streams and output shot-level importance scores.

Our model is different from MSVA as MSVA treats each feature separately, while we perform early fusion and concatenate representations from multiple backbones even before obtaining a compact video shot representation. TaleSumm is also developed for encoding and making predictions on multiple modalities, while MSVA is not.

PGL-SUM [103] splits the video in small equal-sized group of frames. Similar to our work, contextualization is performed within local multi-headed self-attention on small groups, while another is at global level using global multi-headed attention for the entire video. Later, both are merged via addition along the feature axis and subsequently passed through an MLP classification head to obtain frame-level scores. To adapt PGL-SUM to our work, we again think of shots as the basic unit and concatenate visual features from all streams ($\mathbf{f}^1, \mathbf{f}^2$, and $\mathbf{f}^3$), followed by pooling (ST or Max or Avg pooling) and then PGL-SUM to generate shot-level importance scores.

PGL-SUM has some similarities to our approach as both involve local groups. Interestingly, PGL-SUM creates groups of frames to perform summarization for short videos of a few minutes, while TaleSumm creates groups of shots and dialog utterances to generate summaries for 40 minute long episodes. Among technical contributions, we also explore different attention mechanisms such as across the full-episode (FE) or within a local story group (SG). Different from PGL-SUM, we introduce a story group token that allows to capture the essence of a local story group.

PreSumm [35] is used for text-only extractive summarization which takes word-level inputs and produces sentence-level probability scores. To represent each episode, the utterances are concatenated, lower-cased, and separated by CLS and SEP tokens into a single line input. The PreSumm model leverages word embeddings from pre-trained BERT-base [89] language model. Considering the long inputs in our case, we extend the existing positional embeddings of BERT from 512 to 10000 by keeping the original embeddings and replicating the last embeddings for the remainder. At the sentence level, the corresponding CLS token is fed into two transformer encoder layers for contextualization, followed by a small MLP and sigmoid operation to generate per-sentence scores. The model is trained using the Adam [105] optimizer with Binary Cross-Entropy loss.

PreSumm is very different from our work as it operates directly on tokens, while our model develops a hierarchical approach going from words to dialogs to local story groups.

A2Summ [12] is a contemporary multimodal summarization (MSMO) baseline, primarily designed to align temporal correspondence between video and text signals through a dual-contrastive loss approach. They also introduce a dataset, BLiSS [12], comprising 13,303 pairs of livestream videos and transcripts, each with an average duration of 5 minutes, along with multimodal (VT2VT) summaries. A2Summ exploits cross-modality correlations within and between videos through dual contrastive losses. These include: (a) inter-sample contrastive loss (operates across different sample pairs within a batch, leveraging the intrinsic correlations between video-text pairs), and (b) an intra-sample contrastive loss (works

Model	SumMe			TVSum		
	F1	SP	KT	F1	SP	KT
MSVA [104]	52.4	12.3	9.2	*63.9*	32.1	22.0
PGLSUM [103]	56.2	17.3	12.7	*63.9*	**40.5**	**28.2**
A2Summ [12]	54.0	3.5	2.8	62.9	25.2	17.1
TaleSumm (Ours)	**57.5**	**23.8**	**17.6**	**64.0**	26.7	18.2

Table 4.6: Comparison with SoTA methods on the SumMe [1] and TVSum [2] benchmark datasets. Metrics are suggested by Otani *et al*. [3]: F1, Kendall's τ (KT), and Spearman's ρ (SP).

within each sample pair, emphasizing the similarities between ground-truth video and text summaries while contrasting positive features with hard-negative features).

We adapt A2Summ for PlotSnap, where we work at the episodic level, by using max-pooling with an MLP for video features and average-pooling for dialog (text) features to derive shot- and utterance-level representations. We create explicit intra-sample attention masks encouraging temporal alignment, allowing video shots to attend to their corresponding utterances and permitting video and utterance tokens to fully attend to their respective counterparts. Considering memory constraints, we maintain a batch size of 4 (four entire episodes - inter-sample contrasting) on a single NVIDIA GeForce RTX-2080 Ti GPU. We adopt CyclicLR [106] with a maximum learning rate of 10^{-4} and the 'triangular2' mode. A2Summ model comprises 6 encoder layers and incorporates multiple dropout layers [102]: (a) `dropout_video` $=0.1$, (b) `dropout_text` $=0.2$, (c) `dropout_attn` $=0.3$, and (d) `dropout_fc` $=0.5$, while keeping rest the of the hyperparameters the same. Training extends to a maximum of 50 epochs, with the AdamW [4] optimizer utilized, featuring a learning rate of 10^{-5} and a weight decay [107] $=0.01$.

In PlotSnap, where episodes show related content, the application of inter-episode contrastive learning negatively impacts the model's performance. This is due to the variability in the importance of related story segments across episodes, which depends on the specific context.

4.4 Analysis and Discussion

4.4.1 Video summarization benchmarks

We evaluate TaleSumm on SumMe [1] and TVSum [2]. However, both datasets are small (25 and 50 videos) and have short duration videos (few minutes). As splits and metrics are not comparable across previous works, we re-ran the baselines.

While MSVA uses three feature sets: i3d-rgb, i3d-flow [108] and GoogleNet [109] with intermediate fusion, PGLSUM uses GoogleNet and captures local and global features. In contrast, A2Summ [12] aligns cross-modal information using dual-contrastive loss between video (GoogleNet features) and text (captions generated using GPT-2 [110], embedded by RoBERTa [97] at frame level).

Similar to MSVA, we fuse all 3 features. Even though TaleSumm is built for long videos (group blocks, sparse attention), Table 4.6 shows that we achieve SoTA on SumMe. The drop in performance on TVSum may be due to diverse genres (documentaries, how-to videos, *etc*.).

4.4.2 Adaptation of TaleSumm for SumMe and TVSum

In this section, we will discuss how we adapted our model for SumMe [1] and TVSum [2], along with the comprehensive hyperparameter details and the corresponding evaluation metrics.

Adaptation. We used our video-based model on both datasets, inheriting features from MSVA[3] [104]. To capture shot-level details, we stacked 15 contiguous frame-embeddings (as previous methods utilized ground-truth labels indexed at every 15th frame), and for group-level, we used `n_frame_per_seg` attribute of the dataset. We used continuous-index-based time embeddings for shot-frames. Essentially, we assume that a shot consists of 15 frames as SumMe and TVSum require predictions at every 15 frames.

Hyperparameter configuration. We determine the configuration based on the best validation score obtained over five random splits (5-RCV). This time our model is trained on a single NVIDIA GeForce RTX-2080 Ti GPU for a maximum of 300 epochs for SumMe and 100 epochs for TVSum, with a batch size of 1. Additional dataset-specific hyperparameters are detailed in Table 4.7. In common, we have AdamW optimizer [4] with parameters: learning rate $=5\times10^{-5}$, weight decay [107] $=10^{-3}$. We use

[3] https://github.com/TIBHannover/MSVA/tree/master

	amsgrad	d_model	drop_fc/trm/proj	dec_l	enc_l	wd	act_clf/mlp/trm	ffs
SumMe [1]	True	512	0.5/0.2/0.2	3	1	0.0001	r/g/g	Stack
TVSum [2]	False	768	0.7/0.4/0.2	6	1	0.01	r/g/g	⊞

Table 4.7: Hyperparameter configuration for SumMe and TVSum. `d_model` specify the dimension for the transformer module's internal representation, while `dec_l` and `enc_l` denote # of decoder and encoder layers, respectively. Other hyperparameters include `drop_fc/trm/proj` for dropout at classification head, attention module, and projection module, `wd` stands for weight decay parameter (used inside AdamW [4]), `act_clf/mlp/trm` for activation function used in classification head, projection, and attention module, with `r` for ReLU and `g` for GELU. Lastly, `ffs` stands for feature fusion style, with Stack and ⊞ depicting the pooling strategy showed in Table 4.2.

	Model	X-Season (*24*)		X-Series (*PB*)	
		Video AP	Dialog AP	Video AP	Dialog AP
1	MSVA [104]	46.7 ± 2.7	-	32.7	-
2	PGLSUM [103]	47.1 ± 2.4	-	34.5	-
3	PreSumm [35]	-	41.3 ± 3.2	-	**38.3**
4	A2Summ [12]	33.5 ± 3.2	31.7 ± 2.9	20.2	19.0
5	TaleSumm (Ours)	**51.0** ± 4.6	**46.0** ± 5.5	**36.7**	*35.7*

Table 4.8: We evaluate our model's generalization across seasons within *24* and across TV shows (*24* $\rightarrow$ *Prison Break*). R5 showcases superiority of our methods compared to SoTA. In X-Series, the random baseline achieves 21.3 and 19.1 for video and dialog.

CyclicLR [106] for learning rate scheduling with max lr $=5\times10^{-4}$ and *triangular2* mode. ReLU is used for classification head and GELU for projection and attention modules.

Generalization to a new season/TV series. Table 4.8 shows results in two different setups. In X-Season, we see the impact of evaluating on unseen seasons (in a 7-fold cross-val-test). While TaleSumm outperforms baselines, it is interesting that most methods show comparable performance across IntraCVT and X-Season setups (see Tables 4.5 and 4.8).

In the X-Series setting, we train our model on *24*and evaluate on *Prison Break*. While both series are

Dataset	Cronbach α		Pairwise F_1		Fleiss' κ	
	Video	Dialog	Video	Dialog	Video	Dialog
SumMe	0.88	-	0.31	-	0.21	-
TVSum	0.98	-	0.36	-	0.15	-
PlotSnap (Ours)	0.91	0.93	0.59	0.60	0.38	0.39

Table 4.9: Label consistency across datasets.

Methods	Fandom (F)		Human (H)	
	Video AP	Dialog AP	Video AP	Dialog AP
GT	64.1	63.8	44.8	42.2
PGLSUM	43.0	-	47.6	-
MSVA	34.3	-	42.2	-
PreSumm	-	43.6	-	46.7
A2Summ	28.7	29.7	41.0	41.1
TaleSumm	**44.5**	**45.5**	**48.7**	**50.9**

Table 4.10: Results on labels from *24* fan site (F) and human-annotated story summaries (H) averaged over 17 episodes of *24*.

crime thrillers, there are significant visual and editing differences between the two shows. Our approach obtains good scores on video summarization, and is a close second on dialog.

4.4.3 Label consistency.

As suggested by [1, 2], label consistency is crucial for evaluating summarization methods. We assess the consistency of labels in PlotSnap using Cronbach's α, pairwise F_1-measure, and a new agreement score: Fleiss' κ. We utilize three labelers for 17 episodes of *24*. (i) GT: obtained from matching recaps; (ii) F: maps plot events as text descriptions to videos from a *24* fan site[4]; (iii) H: human response for a summary. Our labels have superior consistency compared to SumMe [1] and TVSum [2] (see Table 4.9),

[4]https://24.fandom.com/wiki/Day_6:_4:00am-5:00am talks about the key story events in S06E22 in a *Previously on 24* section (see Fig. 4.3).

indicating that identifying key story events in a TV episode is less subjective than scoring importance for generic Youtube videos. Table 4.10 shows results for baselines and TaleSumm on the two other labels F and H. Our model predictions align better with both other labels. Despite fair agreement between human and ground-truth labels, our model's predictions strongly align with human preferences, highlighting the quality of the generated summary.

Further, Table 4.11 expands on the individual scores obtained for each of the 17 episodes. We observe that Cronbach's α is consistently high, while Fleiss's κ varies typically between 0.2–0.5 indicating fair to moderate agreement.

season	episode	Video			Dialog		
		Cα	PF_1	Fκ	Cα	PF_1	Fκ
S02	E21	0.978	0.51	0.337	0.964	0.51	0.335
S02	E23	0.937	0.407	0.135	0.966	0.472	0.182
S03	E20	0.959	0.684	0.595	0.939	0.715	0.629
S03	E22	0.907	0.568	0.422	0.861	0.553	0.409
S04	E20	0.954	0.671	0.47	0.949	0.716	0.568
S04	E21	0.981	0.618	0.405	0.983	0.504	0.286
S05	E21	0.886	0.521	0.27	0.872	0.497	0.249
S05	E22	0.994	0.573	0.282	0.991	0.611	0.334
S06	E20	0.986	0.639	0.432	0.975	0.689	0.534
S06	E21	0.979	0.645	0.437	0.96	0.618	0.439
S06	E22	0.965	0.557	0.297	0.929	0.632	0.406
S06	E23	0.981	0.496	0.263	0.986	0.475	0.206
S07	E20	0.942	0.684	0.451	0.968	0.67	0.382
S07	E22	0.849	0.531	0.349	0.892	0.54	0.334
S07	E23	0.993	0.525	0.215	0.988	0.541	0.235
S08	E21	0.233	0.68	0.527	0.715	0.667	0.518
S08	E22	0.968	0.685	0.507	0.948	0.728	0.538
Average		0.911	0.588	0.376	0.934	0.596	0.387

Table 4.11: Detailed overview of reliability scores for 17 episodes from *24* (test set of *default-split*) with the last row showing the average across all episodes. *Si* and *Ej* stands for Season i Episode j, Cα for Cronbach's α, PF_1 for Pairwise F_1, and Fκ for Fleiss' κ.

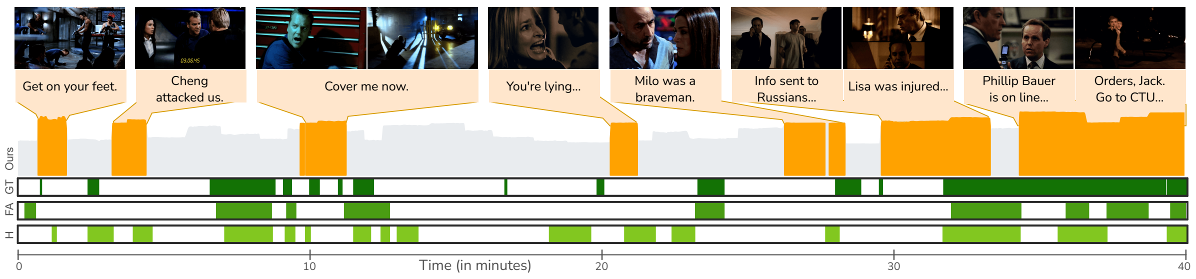

Figure 4.3: TaleSumm predictions on S06E22 of *24* (test set). "Ours" filled-plot illustrates the importance score profile over time, where orange patches indicate story segments selected for summarization. Annotations are shown below: ground-truth (GT), fandom (F), and human annotated (H).

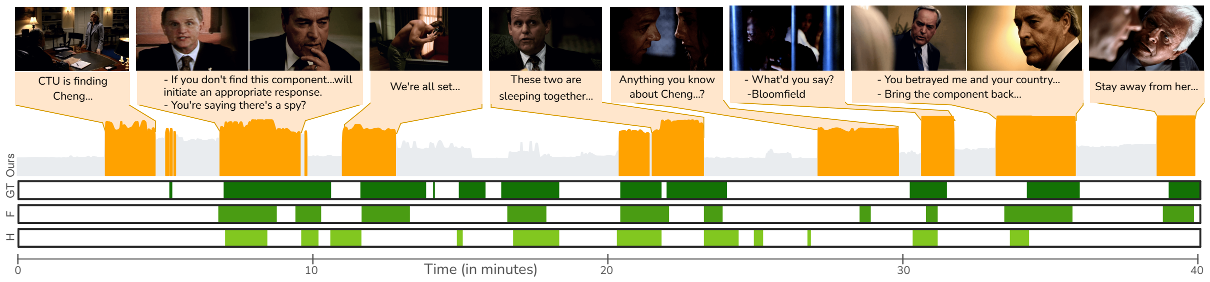

Figure 4.4: TaleSumm predictions on S06E20 of *24* (test set). "Ours" filled-plot illustrates the importance score profile over time, where orange patches indicate story segments selected for summarization. Annotations are shown below: ground-truth (GT), fandom (F), and human annotated (H).

4.4.4 Qualitative Analysis

In this section, we analyze 4 episodes and compare the model's prediction against all three labels (GT, F, and H). The labels denoted *F* (Fandom) are based on summarized plot synopses from the *24* fan site[5] that includes the key events in the story (as a text description). Plot synopses are short textual descriptions of the key story events of an episode. We ask annotators to use the plot synopses and tag the start-end duration for story sequences corresponding to the description. We refer to these labels as *Fandom* (F) and use them for qualitative evaluation. While the *H*-labels are annotations from a human, based on what they feel is relevant summary as per the narrative.

Qualitative evaluation. We present importance scores for four episodes in Fig. 4.3, Fig. 4.4, Fig. 4.5, and Fig. 4.6, respectively:

1. S06E22[4]: We number the grouped frames representing the predicted contiguous orange chunks as shot groups (SG-n), e.g. this episode has 7 SGs.

[5]https://24.fandom.com/wiki/Wiki_24

The story: Amid the high-stakes sequence depicted in the selected groups SG-1,2,3, Zhou Yong's team captures Josh Bauer, leading to a firefight with Jack Bauer, who seeks Josh's location. Negotiations with Phillip Bauer over Josh's return for a vital circuit board escalate global tensions between Russia and the USA. Simultaneously, Mike Doyle defies Jack's wishes and departs with Josh by helicopter (SG-7). Parallely, Lisa, backed by Tom Lennox, confronts a Russian agent, leading to her injury (SG-4,6). Morris attempts to console Nadia for Milo's loss at CTU in SG-5. Escalating global tensions and the imminent showdown mark the episode.

2. S06E20[6]: Here again we number the grouped frames into contiguous orange chunks as shot groups (SG-n), *e.g.* this episode has 8 SGs. **The story**: The White House directs CTU to locate Cheng, as depicted in SG-1, who possesses a Russian sub-circuit board that threatens national security. In SG-2, President Suvarov warns of military consequences if the Chinese agent with the circuit board isn't intercepted. SG-3,4,7 shows how Lennox suspects a spy within the administration and uncovers Lisa's treason. President Noah Daniels instructs Lisa to bring the component back by misleading her partner, Mark Bishop. In SG-5,6, Jack questions Audrey about Cheng, leading to a standoff with Doyle. Audrey mentions "Bloomfield," prompting research by Chloe. In his holding room, Heller warns Jack to stay away from Audrey due to the deadly consequences associated with him (SG-8). Intricate relationships and the imminent threat of international conflict mark the overall content of this episode.

3. S07E22[7]: Here $n = 8$, *e.g.* this episode has 8 SGs. **The story**: In a tense sequence, as shown in SG-1, Jack resorts to torture to extract information from Harbinson about the impending attack but is left empty-handed. In SG-2, following the murder of Jonas Hodges, Olivia Taylor faces scrutiny from the Justice Department. Meeting with Martin Collier, she denies transferring funds, revealing a sinister plot. Meanwhile, SG-3 shows Kim Bauer's plans are disrupted by a flight delay, leading to a strained father-daughter relationship. SG-4,5 displays how Jack, aided by Chloe O'Brian and Renee Walker, captures Tony Almeida and interrogates him about a dangerous canister, followed by Renee uncovering Jibraan's location, and a high-stakes exchange ensues at the Washington Center station. Jack detonates the canister, succumbing to its effects. As a consequence (SG-6), Cara Bowden reports Tony's failure to Alan Wilson, adding tension to the unfolding crisis. Olivia returns

[6] `https://24.fandom.com/wiki/Day_6:_2:00am-3:00am` talks about the key story events in S06E20 in a *Previously on 24* section (see Fig. 4.4).

[7] `https://24.fandom.com/wiki/Day_7:_6:00am-7:00am` talks about the key story events in S07E22 in a *Previously on 24* section (see Fig. 4.5).

to the White House, explaining her absence to Aaron Pierce (SG-7), which beautifully connects back to the SG-2. The narrative takes a dire turn as Cara blackmails Jack for the safety of Kim (SG-8), introducing a new layer of suspense and complexity to the unfolding events. *The presence of SG-1,6,7 (absent in GT) clearly highlights our model's ability to complete the overall story arc.*

4. S05E21[8]: Here we have $n = 5$, *e.g.* this episode has 5 SGs. This episode stands out due to its rapid and significant story advancements, where each sub-story holds apparent importance. Also, human annotations are a bit off in comparison to the ground-truth (can be verified from reliability scores shown in Table 4.11). Importantly, our model considers opinions from all the sources. **The story**: In the SG-1,2, President Logan, pretending surprised, learns from Admiral Kirkland that Flight 520, now under Jack's control, is a potential threat. Despite Mike's doubts about Jack's intentions, Kirkland urges immediate action, advocating for shooting down the plane. Logan, pretending shock, reluctantly authorizes the attack. Karen alerts Jack to the order, leading to a tense situation. As the plane assumes a landing profile, Kirkland suggests calling off the strike, but Pres. Logan insists on taking it down. Further, in SG-3, Graem criticizes Logan's decision, emphasizing the importance of capturing Jack, but Logan assures Graem of recapturing him. Meanwhile, in SG-4, Jack, having secured incriminating evidence, vows to make Logan pay for President Palmer's assassination. In a surprising turn, as shown in SG-5, President Logan contemplates suicide, but an unexpected call from Miles Papazian presents an alternative – the destruction of the recording. Encouraging Miles to act, Logan faces a critical juncture in the unfolding crisis. Overall the entire episode sets the stage for a series of dramatic events, stressing the depth of deceit and the potential consequences for key characters.

We observe that the model predictions are quite good and not only match the ground-truth labels (on which the model is trained) but also the fandom and human annotations. Please refer to the figure captions for additional comments on episode-specific remarks.

4.5 Limitations and Future Work

Our approach considers coarse-grained visual information, which we demonstrate is beneficial for story-summarization. Considering more fine-grained visual info, such as person and face tracks across

[8]`https://24.fandom.com/wiki/Day_5:_4:00am-5:00am` talks about the key story events in S05E21 in a *Previously on 24* section (see Fig. 4.6).

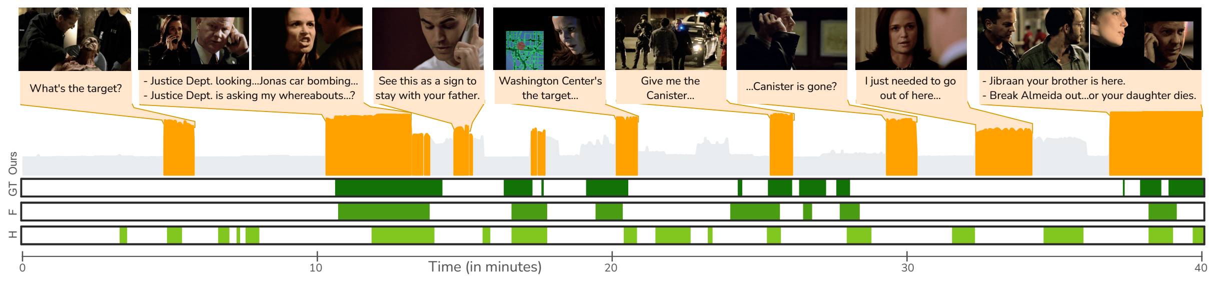

Figure 4.5: TaleSumm predictions on S07E22 of *24* (test set). "Ours" filled-plot illustrates the importance score profile over time, where orange patches indicate story segments selected for summarization. Annotations are shown below: ground-truth (GT), fandom (F) and human annotated (H).

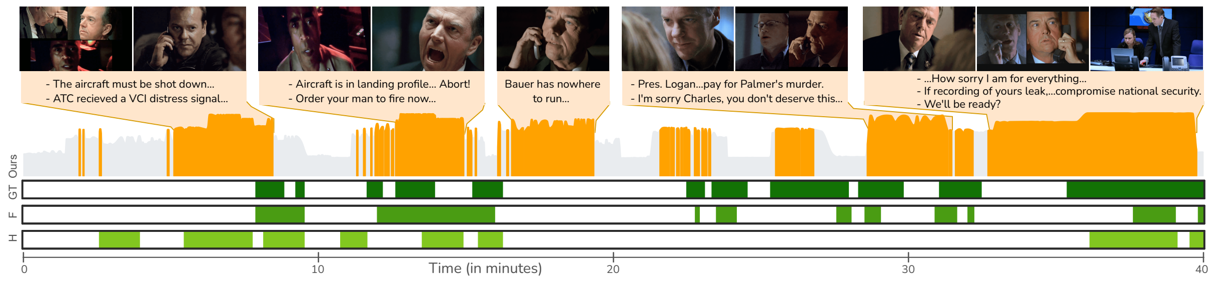

Figure 4.6: TaleSumm predictions on S05E21 of *24* (test set). "Ours" filled-plot illustrates the importance score profile over time, where orange patches indicate story segments selected for summarization. Annotations are shown below: ground-truth (GT), fandom (F), and human annotated (H).

frames, and their emotions, would be useful. Further, having speaker information in dialog utterances with mapping to character faces would probably improve performance on the summarization task.

The local story groups are a proxy to scene segments of an episode. Replacing them with actual scene segments may improve our model's performance for summarization.

Besides all these, ingesting ∼40 minute long videos is a challenging problem. Aligning different modalities may require a larger GPU memory.

Finally, further analysis and experiments are required to determine the quality of these methods [111], particularly because evaluating long video summarization using human judgment is very time-consuming.

However, we believe that this work provides a window into this challenging problem and can help facilitate further research in this area.

Chapter 5

Conclusion

Our work is the first to use summaries of TV episodes to help understand their stories better. We proposed a dataset named, PlotSnap containing high-quality recaps for 2 TV shows, *24*and *Prison Break*. We used these recaps to label the key points of the storyline of an TV Show episode for summarization. We also developed a hierarchical summarization method, TaleSumm, that, at first level, captures and compresses both visual and textual information from each shot or conversation, respectively, in the episode. In the second, it allows cross-modal interactions across different signals to figure out which parts are essential to form story-summary. We performed thorough experiments via ablations to see what works best. Our method exhibits SoTA performance in comparison to other method before. Additionally, we analyzed it transferability across seasons/series, and ensured that our labels for summarizing the story were consistent across different episodes. For code, dataset (including key images, features, and labels), and implementation instruction, follow our repository[1]. As well as to see more demo and examples, kindly check out our paper and project web-page (mentioned in GitHub repository itself).

[1] https://github.com/katha-ai/RecapStorySumm-CVPR2024

Bibliography

[1] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *European Conference on Computer Vision Workshops (ECCVW)*. Springer, 2014, pp. 505–520.

[2] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes, "Tvsum: Summarizing web videos using titles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5179–5187.

[3] M. Otani, Y. Nakashima, E. Rahtu, and J. Heikkilä, "Rethinking the evaluation of video summaries," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 7588–7596.

[4] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," in *International Conference on Learning Representations (ICLR)*, 2017.

[5] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," in *International Conference on Machine Learning (ICML)*, 2019.

[6] P. Papalampidi and M. Lapata, "Hierarchical3D Adapters for Long Video-to-text Summarization," *arXiv:2210.04829*, 2022.

[7] R. Sanabria, O. Caglayan, S. Palaskar, D. Elliott, L. Barrault, L. Specia, and F. Metze, "How2: A Large-scale Dataset for Multimodal Language Understanding," in *Advances in Neural Information Processing Systems-Workshop (NeurIPS-W)*, 2018.

[8] M. Narasimhan, A. Rohrbach, and T. Darrell, "Clip-It! Language-guided Video Summarization," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[9] A. Kulesza, B. Taskar, *et al.*, "Determinantal point processes for machine learning," *Foundations and Trends® in Machine Learning*, vol. 5, no. 2–3, pp. 123–286, 2012.

[10] J. Zhu, H. Li, T. Liu, Y. Zhou, J. Zhang, and C. Zong, "MSMO: Multimodal summarization with multimodal output," in *Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4154–4164. [Online]. Available: https://aclanthology.org/D18-1448

[11] X. Fu, J. Wang, and Z. Yang, "MM-AVS: A full-scale dataset for multi-modal summarization," in *North American Chapter of Association of Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Online: Association for Computational Linguistics, June 2021, pp. 5922–5926. [Online]. Available: https://aclanthology.org/2021.naacl-main.473

[12] B. He, J. Wang, J. Qiu, T. Bui, A. Shrivastava, and Z. Wang, "Align and attend: Multimodal summarization with dual contrastive losses," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[13] W. Wolf, "Key Frame Selection by Motion Analysis," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, ser. ICASSP '96. USA: IEEE Computer Society, 1996, p. 1228–1231. [Online]. Available: https://doi.org/10.1109/ICASSP.1996.543588

[14] Y. J. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1346–1353.

[15] A. Khosla, R. Hamid, C.-J. Lin, and N. Sundaresan, "Large-Scale Video Summarization Using Web-Image Priors," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[16] G. Kim, L. Sigal, and E. P. Xing, "Joint Summarization of Large-Scale Collections of Web Images and Videos for Storyline Reconstruction," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 4225–4232.

[17] Z. Lu and K. Grauman, "Story-Driven Summarization for Egocentric Video," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2714–2721.

[18] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *European Conference on Computer Vision Workshops (ECCVW)*. Springer, 2014, pp. 505–520.

[19] D. B. Goldman, B. Curless, D. Salesin, and S. M. Seitz, "Schematic Storyboarding for Video Visualization and Editing," *ACM Transactions on Graphics*, vol. 25, no. 3, p. 862–871, jul 2006. [Online]. Available: https://doi.org/10.1145/1141911.1141967

[20] J. Kopf, M. F. Cohen, and R. Szeliski, "First-Person Hyper-Lapse Videos," *ACM Transactions on Graphics*, vol. 33, no. 4, jul 2014. [Online]. Available: https://doi.org/10.1145/2601097.2601195

[21] M. Sun, A. Farhadi, B. Taskar, and S. Seitz, "Salient montages from unconstrained videos," in *European Conference on Computer Vision Workshops (ECCVW)*. Springer, 2014, pp. 472–488.

[22] Y. Pritch, A. Rav-Acha, and S. Peleg, "Nonchronological Video Synopsis and Indexing," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 30, no. 11, 2008, pp. 1971–1984.

[23] V. V. K., D. Sen, and B. Raman, "Video skimming: Taxonomy and comprehensive survey," *ACM Comput. Surv.*, vol. 52, no. 5, sep 2019. [Online]. Available: https://doi.org/10.1145/3347712

[24] M. Gygli, H. Grabner, and L. Van Gool, “Video summarization by learning submodular mixtures of objectives,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3090–3098.

[25] K. Zhang, W. Chao, F. Sha, and K. Grauman, “Summary transfer: Exemplar-based subset selection for video summarization,” *CoRR*, vol. abs/1603.03369, 2016. [Online]. Available: http://arxiv.org/abs/1603.03369

[26] M. Ma, S. Mei, S. Wan, Z. Wang, D. D. Feng, and M. Bennamoun, “Similarity based block sparse subset selection for video summarization,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3967–3980, 2021.

[27] W. Zhu, J. Lu, J. Li, and J. Zhou, “DSNet: A Flexible Detect-to-Summarize Network for Video Summarization,” *IEEE Transactions on Image Processing*, vol. 30, pp. 948–962, 2021.

[28] W. Zhu, Y. Han, J. Lu, and J. Zhou, “Relational reasoning over spatial-temporal graphs for video summarization,” *IEEE Transactions on Image Processing*, vol. 31, pp. 3017–3031, 2022.

[29] K. Zhang, K. Grauman, and F. Sha, “Retrospective encoders for video summarization,” in *European Conference on Computer Vision Workshops (ECCVW)*, 2018, pp. 383–399.

[30] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, “Video summarization with long short-term memory,” in *European Conference on Computer Vision Workshops (ECCVW)*. Springer, 2016, pp. 766–782.

[31] Y. Zhang, M. Kampffmeyer, X. Zhao, and M. Tan, “DTR-GAN: Dilated temporal relational adversarial network for video summarization,” in *Proceedings of the ACM Turing Celebration Conference-China*, 2019, pp. 1–6.

[32] T.-J. Fu, S.-H. Tai, and H.-T. Chen, “Attentive and adversarial learning for video summarization,” in *Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1579–1587.

[33] M. Rochan and Y. Wang, “Video summarization by learning from unpaired data,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 7902–7911.

[34] G. Awad, K. Curtis, A. A. Butt, J. Fiscus, A. Godil, Y. Lee, A. Delgado, J. Zhang, E. Godard, B. Chocot, L. Diduch, J. Liu, Y. Graham, , and G. Quénot, “An overview on the evaluated video retrieval tasks at TRECVID 2022,” in *Proceedings of TRECVID*, 2022.

[35] Y. Liu and M. Lapata, “Text summarization with pretrained encoders,” in *Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3730–3740. [Online]. Available: https://aclanthology.org/D19-1387

[36] M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang, "Extractive summarization as text matching," in *Association of Computational Linguistics (ACL)*. Online: Association for Computational Linguistics, July 2020, pp. 6197–6208. [Online]. Available: https://aclanthology.org/2020.acl-main.552

[37] R. Jia, Y. Cao, H. Tang, F. Fang, C. Cao, and S. Wang, "Neural extractive summarization with hierarchical attentive heterogeneous graph network," in *Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 3622–3631. [Online]. Available: https://aclanthology.org/2020.emnlp-main.295

[38] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Association of Computational Linguistics (ACL)*. Association for Computational Linguistics, 2020.

[39] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research (JMLR)*, 2020.

[40] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," in *International Conference on Machine Learning (ICML)*. PMLR, 2020, pp. 11 328–11 339.

[41] R. Nallapati, B. Zhou, C. dos Santos, Çaglar Gulçehre, and B. Xian, "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond," in *Computational Natural Language Learning (CoNLL)*, 2016.

[42] P. Papalampidi, F. Keller, L. Frermann, and M. Lapata, "Screenplay Summarization Using Latent Narrative Structure," in *Association of Computational Linguistics (ACL)*, 2020.

[43] M. Chen, Z. Chu, S. Wiseman, and K. Gimpel, "SummScreen: A dataset for abstractive screenplay summarization," in *Association of Computational Linguistics (ACL)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 8602–8615. [Online]. Available: https://aclanthology.org/2022.acl-long.589

[44] P. Papalampidi, F. Keller, and M. Lapata, "Movie plot analysis via turning point identification," in *Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 1707–1717. [Online]. Available: https://aclanthology.org/D19-1180

[45] ——, "Movie Summarization via Sparse Graph Construction," in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2021.

[46] A. Rohrbach, M. Rohrbach, W. Qiu, A. Friedrich, M. Pinkal, and B. Schiele, “Coherent Multi-sentence Video Description with Variable Level of Detail,” in *German Conference on Pattern Recognition (GCPR)*, 2014.

[47] J.-H. Huang and M. Worring, “Query-controllable Video Summarization,” in *International Conference on Multimedia Retrieval (ICMR)*, 2020.

[48] Y. Saquil, D. Chen, Y. He, C. Li, and Y.-L. Yang, “Multiple Pairwise Ranking Networks for Personalized Video Summarization,” in *International Conference on Computer Vision (ICCV)*, 2021.

[49] P. Papalampidi, F. Keller, and M. Lapata, “Film trailer generation via task decomposition,” *arXiv preprint arXiv:2111.08774*, 2021.

[50] S. E. F. de Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo, “VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method,” *Pattern Recognition Letters*, vol. 32, no. 1, pp. 56–68, 2011.

[51] R. Nallapati, B. Zhou, C. Gulcehre, and B. Xiang, “Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond,” in *Computational Natural Language Learning (CoNLL)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 280–290. [Online]. Available: https://aclanthology.org/K16-1028

[52] S. Narayan, S. B. Cohen, and M. Lapata, “Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization,” in *Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 1797–1807. [Online]. Available: https://aclanthology.org/D18-1206

[53] H.-I. Ho, W.-C. Chiu, and Y.-C. F. Wang, “Summarizing first-person videos from third persons’ points of views,” in *European Conference on Computer Vision Workshops (ECCVW)*. Berlin, Heidelberg: Springer-Verlag, 2018, p. 72–89. [Online]. Available: https://doi.org/10.1007/978-3-030-01267-0_5

[54] K.-H. Zeng, T.-H. Chen, J. C. Niebles, and M. Sun, “Title generation for user generated videos,” in *European Conference on Computer Vision Workshops (ECCVW)*, 2016.

[55] C.-Y. Fu, J. Lee, M. Bansal, and A. Berg, “Video highlight prediction using audience chat reactions,” in *Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark: Association for Computational Linguistics, Sept. 2017, pp. 972–978. [Online]. Available: https://aclanthology.org/D17-1102

[56] B. Gliwa, I. Mochol, M. Biesek, and A. Wawer, “SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization,” in *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 70–79. [Online]. Available: https://aclanthology.org/D19-5409

[57] M.-L. Haurilet, M. Tapaswi, Z. Al-Halah, and R. Stiefelhagen, “Naming tv characters by watching and analyzing dialogs,” in *Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–9.

[58] A. Nagrani, S. Albanie, and A. Zisserman, “Learnable PINs: Cross-Modal Embeddings for Person Identity,” in *European Conference on Computer Vision Workshops (ECCVW)*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:13746939

[59] A. Nagrani and A. Zisserman, “From Benedict Cumberbatch to Sherlock Holmes: Character Identification in TV series without a Script,” in *British Machine Vision Conference (BMVC)*, 2017.

[60] Tengda Han and Max Bain and Arsha Nagrani and Gül Varol and Weidi Xie and Andrew Zisserman, “Autoad II: the sequel - who, when, and what in movie audio description,” in *International Conference on Computer Vision (ICCV)*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2310.06838

[61] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler, “Movieqa: Understanding stories in movies through question-answering,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4631–4640.

[62] J. Lei, L. Yu, M. Bansal, and T. Berg, “TVQA: Localized, compositional video question answering,” in *Empirical Methods in Natural Language Processing (EMNLP)*, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 1369–1379. [Online]. Available: https://aclanthology.org/D18-1167

[63] J. Lei, L. Yu, T. Berg, and M. Bansal, “TVQA+: Spatio-temporal grounding for video question answering,” in *Association of Computational Linguistics (ACL)*, July 2020, pp. 8211–8225. [Online]. Available: https://aclanthology.org/2020.acl-main.730

[64] A. Rohrbach, M. Rohrbach, N. Tandon, and B. Schiele, “A dataset for Movie Description,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3202–3212.

[65] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. Pal, H. Larochelle, A. Courville, and B. Schiele, “Movie Description,” *International Journal of Computer Vision (IJCV)*, vol. 123, no. 1, p. 94–120, may 2017. [Online]. Available: https://doi.org/10.1007/s11263-016-0987-1

[66] J. S. Park, T. Darrell, and A. Rohrbach, “Identity-Aware Multi-Sentence Video Description,” in *European Conference on Computer Vision Workshops (ECCVW)*, 2020, p. 360–378. [Online]. Available: https://doi.org/10.1007/978-3-030-58589-1_22

[67] S. Pini, M. Cornia, F. Bolelli, L. Baraldi, and R. Cucchiara, “M-VAD Names: A Dataset for Video Captioning with Naming,” *Multimedia Tools Appl.*, p. 14007–14027, 2019. [Online]. Available: https://doi.org/10.1007/s11042-018-7040-z

[68] P. Vicol, M. Tapaswi, L. Castrejon, and S. Fidler, “MovieGraphs: Towards Understanding Human-Centric Situations from Videos,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[69] A. Sadhu, T. Gupta, M. Yatskar, R. Nevatia, and A. Kembhavi, "Visual semantic role labeling for video understanding," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[70] Z. Khan, C. Jawahar, and M. Tapaswi, "Grounded video situation recognition," in *Advances in Neural Information Processing Systems (NeurIPS)*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. [Online]. Available: https://openreview.net/forum?id=yRhbHp_Vh8e

[71] F. Xiao, K. Kundu, J. Tighe, and D. Modolo, "Hierarchical Self-supervised Representation Learning for Movie Understanding," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, jun 2022.

[72] M. Tapaswi, M. Bäuml, and R. Stiefelhagen, "Aligning plot synopses to videos for story-based retrieval," *International Journal of Multimedia Information Retrieval (IJMIR)*, vol. 4, no. 1, pp. 3–16, 2015.

[73] M. Tapaswi, M. Bäuml, and R. Stiefelhagen, "Story-based video retrieval in tv series using plot synopses," in *International Conference on Multimedia Retrieval (ICMR)*, ser. ICMR '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 137–144. [Online]. Available: https://doi.org/10.1145/2578726.2578727

[74] Y. Zhu, R. Kiros, R. S. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books," in *International Conference on Computer Vision (ICCV)*, 2015, pp. 19–27.

[75] K. P. Sankar, C. V. Jawahar, and A. Zisserman, "Subtitle-free Movie to Script Alignment," in *British Machine Vision Conference (BMVC)*, 2009.

[76] M. Islam, M. Hasan, K. Athrey, T. Braskich, and G. Bertasius, "Efficient movie scene detection using state-space transformers," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2023, pp. 18 749–18 758. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.01798

[77] M. M. Islam and G. Bertasius, "Long movie clip classification with state-space video models," in *European Conference on Computer Vision Workshops (ECCVW)*. Springer, 2022, pp. 87–104.

[78] S. Chen, C.-H. Liu, X. Hao, X. Nie, M. Arap, and R. Hamid, "Movies2scenes: Using movie metadata to learn scene representation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [Online]. Available: https://www.amazon.science/publications/movies2scenes-using-movie-metadata-to-learn-scene-representation

[79] X. Hao, K. Chettiar, B. Cheung, V. Germano, and R. Hamid, "Intro and Recap Detection for Movies and TV Series," in *Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[80] "24 (TV series, IMDb)," https://www.imdb.com/title/tt0285331/, 2001.

[81] "Prison Break (IMDb)," https://www.imdb.com/title/tt0455275/, 2005.

[82] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700–4708.

[83] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, pp. 211–252, 2015.

[84] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 770–778. [Online]. Available: https://api.semanticscholar.org/CorpusID:206594692

[85] M. Tapaswi, M. Bäuml, and R. Stiefelhagen, "StoryGraphs: Visualizing Character Interactions as a Timeline," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[86] G. Chéron, J.-B. Alayrac, I. Laptev, and C. Schmid, "A Flexible Model for Training Action Localization with Varying Levels of Supervision," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[87] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.

[88] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[89] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805*, 2018.

[90] L. J. Ba, J. R. Kiros, and G. E. Hinton, "Layer Normalization," *arXiv: 1607.06450*, 2016.

[91] Y. Yusoff, W. J. Christmas, and J. Kittler, "A Study on Automatic Shot Change Detection," in *European Conference on Multimedia Applications, Services and Techniques*, 1998.

[92] N. Yuval, "Reading digits in Natural Images with Unsupervised Feature Learning," in *Advances in Neural Information Processing Systems-Workshop (NeurIPS-W)*, 2011.

[93] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," *cs.toronto.edu*, 2009.

[94] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *International Conference on Computer Vision (ICCV)*, 2021, pp. 6824–6835.

[95] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, A. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," *CoRR*, vol. abs/1705.06950, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:27300853

[96] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning (ICML).* PMLR, 2021, pp. 8748–8763.

[97] L. Zhuang, L. Wayne, S. Ya, and Z. Jun, "A robustly optimized BERT pre-training approach with post-training," in *Proceedings of the 20th Chinese National Conference on Computational Linguistics*. Huhhot, China: Chinese Information Processing Society of China, Aug. 2021, pp. 1218–1227. [Online]. Available: https://aclanthology.org/2021.ccl-1.108

[98] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, *et al.*, "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[99] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "MPNet: Masked and Permuted Pre-Training for Language Understanding," in *Advances in Neural Information Processing Systems (NeurIPS)*, ser. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[100] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 8024–8035.

[101] L. N. Smith and N. Topin, "Super-convergence: very fast training of neural networks using large learning rates," in *Defense + Commercial Sensing*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:260552651

[102] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *ArXiv*, vol. abs/1207.0580, 2012.

[103] E. Apostolidis, G. Balaouras, V. Mezaris, and I. Patras, "Combining global and local attention with positional encoding for video summarization," in *IEEE International Symposium on Multimedia (ISM)*, 2021, pp. 226–234.

[104] J. A. Ghauri, S. Hakimov, and R. Ewerth, "Supervised video summarization via multiple feature sets with parallel attention," *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6s, 2021.

[105] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980*, 2014.

[106] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," in *Winter Conference on Applications of Computer Vision (WACV)*, 2015, pp. 464–472. [Online]. Available: https://api.semanticscholar.org/CorpusID:15247298

[107] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *International Conference on Learning Representations (ICLR)*, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:53592270

[108] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4724–4733.

[109] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[110] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[111] L. Tang, T. Goyal, A. Fabbri, P. Laban, J. Xu, S. Yavuz, W. Kryscinski, J. Rousseau, and G. Durrett, “Understanding factual errors in summarization: Errors, summarizers, datasets, error detectors,” in *Association of Computational Linguistics (ACL)*. Toronto, Canada: Association for Computational Linguistics, 2023, pp. 11 626–11 644. [Online]. Available: https://aclanthology.org/2023.acl-long.650