

Video Summarization with NLP and Vision-Language Models

Bhargava Siva Naga Sai Potluri(bxp230045),
Kavimayil Periyakoravampalayam Komarasamy (kxp230053),
Nikhil Sesha Sai Kondapalli(nxk240025), Sakshi Tokekar(sxt230143)
Group 16
<https://github.com/nikhilkondapalli5/LLMVS-Implementation>

Abstract

We present an implementation of the LLMVS (LLM-based Video Summarization) framework that leverages Large Language Models for video summarization. Our approach uses a Multi-modal LLM to generate textual descriptions of video frames, followed by an LLM to evaluate frame importance within local contexts. A self-attention mechanism aggregates global context for final predictions. Experiments on the TVSum dataset demonstrate superior performance (Kendall’s $\tau=0.177$, Spearman’s $\rho=0.229$) compared to baseline methods, validating the effectiveness of combining semantic understanding with LLM reasoning.

1 Introduction

The exponential growth of video content across digital platforms has created significant challenges in efficient content navigation, search, and retrieval. Millions of videos are uploaded daily, far exceeding human capacity for consumption and manual processing. This information overload necessitates advanced video summarization techniques that can automatically condense lengthy videos into concise summaries while preserving essential content and narrative structure.

Traditional video summarization methods rely heavily on visual features and temporal dynamics, often failing to capture the semantic content and narrative structure of videos. Recent multi-modal approaches integrate both visual and language modalities but still prioritize visual features, with textual data serving primarily as auxiliary information to enhance visual representations.

Inspired by the work of Lee et al. [1] on ”Video Summarization with Large Language Models,” our project introduces the LLMVS framework. Unlike traditional methods, our approach leverages the semantic understanding and contextual reasoning ca-

pabilities of Large Language Models. We employ a Multi-modal Large Language Model (LLaVA-1.5-7B) to generate textual descriptions of video frames, followed by an LLM (Llama-2-13B) [4] to evaluate frame importance within local temporal contexts.

A key innovation in our methodology is the use of LLM output embeddings rather than direct text answers, which allows us to extract richer semantic information. These local importance scores are then refined through a self-attention mechanism that captures global context across the entire video sequence, ensuring summaries effectively reflect both details and the overarching narrative.

We conducted experiments on the TVSum dataset, which contains 50 videos spanning diverse genres including how-to videos, documentaries, and vlogs. Our implementation utilized a train-test split of 40 training videos and 10 test videos. The evaluation employed ranking metrics (Kendall’s τ and Spearman’s ρ) to measure the correlation between predicted and ground truth importance scores. Our results demonstrate that the LLMVS framework achieves superior performance, outperforming both the A2Summ method and the zero-shot LLM baseline, validating the effectiveness of combining textual understanding with LLM reasoning for video summarization tasks.

2 Data

2.1 Dataset Overview

For our implementation, we utilized the TVSum (TV Summary) dataset [3], a well-established benchmark for video summarization research. The TVSum dataset comprises 50 videos with varying durations and content types, making it ideal for evaluating the generalization capabilities of video summarization models.

2.2 Dataset Characteristics

Size and Duration: The dataset contains 50 videos with durations ranging from 2 to 10 minutes, averaging 4 minutes and 11 seconds per video. The content spans diverse genres including how-to videos, documentaries, vlogs, news clips, and various other categories.

Annotations: Each video is annotated by 20 independent raters who assign importance scores on a continuous scale from 0 to 1. Ground truth scores are provided at the segment level by averaging these user annotations. This multi-rater scheme captures the inherent subjectivity in defining keyframes while providing robust ground truth.

2.3 Data Preprocessing

Prior to feature extraction, the videos were segmented into semantically consistent shots using Kernel Temporal Segmentation (KTS). These pre-computed change points serve as the boundaries for our segment-level importance scoring and ensure that the generated summaries respect natural scene transitions rather than arbitrary fixed-time windows.

2.4 Data Structure

Each video in the dataset is accompanied by metadata that includes:

- **picks:** Sampled frame indices used for feature extraction
- **features:** Pre-extracted visual features using GoogLeNet pool5 layer (dimension: 1024)
- **gtscore:** Ground truth importance scores averaged across user annotations
- **n_frame_per_seg:** Number of frames in each video segment
- **change_points:** Start and end frame indices for each segment (derived via KTS)

As an example, Video ID 38 has a length of 94 seconds, contains 2,941 total frames at 30 fps, with 197 sampled feature steps and 20 segments. The dataset’s diversity in content types and the availability of multiple human annotations make it particularly suitable for evaluating the semantic understanding capabilities of our LLM-based approach, as the ground truth reflects human judgment about frame importance rather than purely visual saliency.

3 Methodology

Our methodology is inspired by recent LLM-based video summarization frameworks and consists of three main stages(Figure 1): frame caption generation, local importance scoring, and global context aggregation.

3.1 Frame Caption Generation

First, each video is decomposed into a sequence of frames. A pre-trained multimodal model(LLaVA-1.5-7B) is used to generate a short natural language description for each frame. These captions aim to capture the main action, objects, and scene context present in the frame, effectively translating visual information into the language domain. This transformation enables us to leverage the semantic understanding capabilities of language models for subsequent video analysis.

3.2 Local Importance Scoring

Next, we perform local importance scoring by evaluating each frame within a temporal window of neighboring frames. For a given frame, captions from a fixed-size sliding window are grouped together and provided as input to a language model(Llama-2-13B). The model is prompted to assess the importance of the central frame based on criteria such as narrative relevance, uniqueness, and action intensity.

Instead of relying only on the final numerical output, we extract intermediate language model embeddings that preserve richer semantic information. These embeddings capture the contextual understanding developed by the language model during its evaluation process, encoding both the query context and the model’s assessment in a high-dimensional representation.

3.3 Global Context Aggregation

Finally, to ensure global coherence across the entire video, we aggregate the locally computed representations using a self-attention-based module. This global aggregation step allows the model to capture long-range dependencies and avoid redundant frame selection. The self-attention mechanism enables the model to weigh the importance of each frame relative to all other frames in the video, ensuring that the final summary reflects the overall narrative structure.

The output is a sequence of importance scores, one per frame, which can be used to generate a final video summary by selecting the most informative segments under a length constraint. During training,

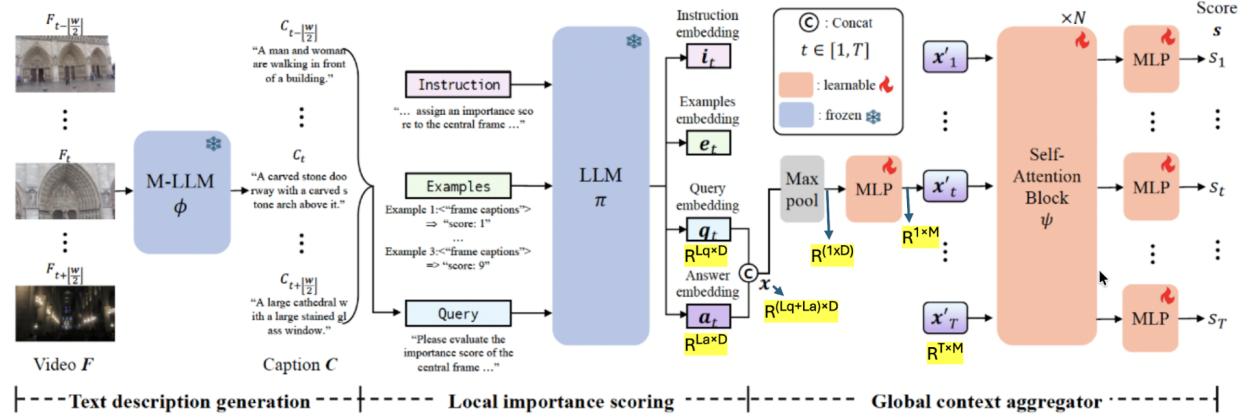


Figure 1: Overall architecture: Input frames are captioned by an M-LLM and passed to an LLM, which estimates frame importance using neighboring context. The resulting embeddings are reduced via an MLP, refined with self-attention to capture global context, and used to produce final importance scores for each frame.

only the global aggregation module is updated, while the pre-trained multimodal and language models remain frozen to preserve their general knowledge.

3.4 Training objective:

The proposed method is trained using the Mean Squared Error (MSE) loss to optimize frame importance predictions. The loss L between the ground truth score vector \hat{s} and the predicted score s and is defined as:

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^T (s_t - \hat{s}_t)^2$$

3.5 Summary Generation

To generate the final video summary, we formulate the selection problem as a 0/1 Knapsack problem. We aim to maximize the total importance score of the selected segments such that their combined duration does not exceed a pre-defined limit (typically 15% of the original video length).

Let v_i be the importance score of segment i , w_i be its duration, and C be the maximum allowed duration. We solve for binary variables $x_i \in \{0, 1\}$ to:

$$\text{maximize} \sum_{i=1}^N v_i x_i \quad \text{subject to} \quad \sum_{i=1}^N w_i x_i \leq C \quad (1)$$

This ensures that the generated summary contains the most semantically significant moments while strictly adhering to the length constraint.

4 Implementation

4.1 Training Infrastructure

Training was conducted on a single NVIDIA A100 GPU using Google Colab, with a training time of 3.5 hours per dataset split. With multiple GPUs, k-fold cross-validation could be implemented using distributed training across all splits simultaneously.

4.2 Architecture Implementation

Our trainable architecture consists of three main components that process the LLM embeddings to produce final importance scores:

Dimension Reduction Module: The 5120-D embeddings from Llama-2-13B are reduced to 2048-D using adaptive max pooling along the sequence dimension, followed by a linear projection and layer normalization for training stability.

Global Context Aggregator: A 3-layer transformer encoder ($d_{\text{model}}=2048$, $n_{\text{head}}=2$) processes the reduced embeddings. Each layer contains multi-head self-attention and feed-forward networks with residual connections and layer normalization. This enables each frame to attend to all other frames, capturing long-range dependencies and global narrative structure.

Score Prediction Head: A 5-layer MLP progressively reduces dimensionality from 2048 to 1, with layer normalization applied at intermediate stages (after $/2$ and $/8$ reductions) for gradient stability. The final sigmoid activation produces importance

scores in the (0,1) range.

The complete PyTorch implementation is shown below:

```

1 # Dimension Reduction Module
2 self.d_max_pooling =
3     nn.AdaptiveMaxPool1d(self.config.reduced_dim)
4 self.d_linear1 = nn.Linear(5120,
5     self.config.reduced_dim)
6 self.d_linear1_norm =
7     nn.LayerNorm(self.config.reduced_dim)
8 self.c_max_pooling = nn.AdaptiveMaxPool1d(1)
9
10 # Global Context Aggregator (Self-Attention
11 # Transformer)
12 encoder_layer_agg = nn.TransformerEncoderLayer(
13     d_model=self.config.reduced_dim,
14     nhead=self.config.num_heads,
15     batch_first=True
16 )
17 self.transformer_encoder_agg =
18     nn.TransformerEncoder(
19         encoder_layer_agg,
20         num_layers=self.config.num_layers
21 )
22
23 # Score Prediction Head (Final MLP)
24 self.mlp_head = torch.nn.Sequential(
25     nn.Linear(self.config.reduced_dim,
26             self.config.reduced_dim//2),
27     nn.LayerNorm(self.config.reduced_dim // 2),
28     nn.ReLU(),
29     nn.Linear(self.config.reduced_dim//2,
30             self.config.reduced_dim//4),
31     nn.ReLU(),
32     nn.Linear(self.config.reduced_dim//4,
33             self.config.reduced_dim//8),
34     nn.LayerNorm(self.config.reduced_dim // 8),
35     nn.ReLU(),
36     nn.Linear(self.config.reduced_dim//8,
37             self.config.reduced_dim//16),
38     nn.ReLU(),
39     nn.Linear(self.config.reduced_dim//16, 1),
40     nn.Sigmoid()
41 )
42
43

```

4.3 Optimization Strategy

The model is trained end-to-end using MSE loss between predicted and ground truth importance scores. We employ the AdamW optimizer with adaptive per-parameter learning rates and decoupled weight decay for improved generalization.

```

1 def configure_optimizers(self):
2     optimizer = torch.optim.AdamW(self.parameters(),
3         lr=self.config.lr)
4     lr_scheduler = {
5         'scheduler':
6             torch.optim.lr_scheduler.CosineAnnealingLR(
7                 optimizer, T_max=100, eta_min=1e-6
8             ),
9         'interval': 'epoch',
10        'frequency': 1,
11    }
12    return [optimizer], [lr_scheduler]

```

4.4 Hyperparameters

- Reduced dimension: 2048

- Attention heads: 2
- Transformer layers: 3
- Training epochs: 200
- Learning rate: 7e-5
- Optimizer: AdamW
- Batch size: 1

Note that only the trainable components (dimension reduction, transformer encoder, and MLP head) are updated during training. The pre-trained LLaVA-1.5-7B and Llama-2-13B models remain frozen to preserve their learned semantic understanding.

5 Experiments and Results

5.1 Data Split

For our experiments, we adopted the following data partitioning strategy: 40 videos (80%) for training and 10 videos (20%) for testing. This split allows adequate training data while maintaining sufficient test samples for a reliable performance evaluation.

5.2 Importance Score Evaluation

We evaluated the quality of predicted importance scores by measuring agreement between the induced ranking of segments and the reference ranking using Kendall's τ and Spearman's ρ (higher is better). Kendall's τ reflects pairwise ordering consistency, while Spearman's ρ measures global monotonic agreement between ranked lists. Baselines marked with * follow the methods reported in the reference work [1].

Table 1: Comparison of importance scores

Method	Kendall's τ	Spearman's ρ
A2Summ*	0.137	0.165
LLM*	0.051	0.056
Our Approach	0.177	0.229

*Methods are based on the reference paper [1]

Results. As shown in Table 1, our approach achieves the strongest alignment with the reference ranking ($\tau = 0.177$, $\rho = 0.229$), outperforming A2Summ* ($\tau = 0.137$, $\rho = 0.165$) and LLM* ($\tau = 0.051$, $\rho = 0.056$).

Analysis. Our method improves over A2Summ* by +0.040 in Kendall’s τ and +0.064 in Spearman’s ρ , indicating fewer pairwise inversions and better overall rank consistency. The gains over LLM* are larger (+0.126 in τ and +0.173 in ρ), suggesting that our design produces more reliable and globally consistent importance rankings than a direct LLM-based baseline from [1]. While absolute correlations remain moderate, the consistent improvement across both rank metrics demonstrates that our approach better captures the relative importance structure needed for summarization.

5.3 Response-level Video Summary Evaluation

In addition to importance-rank agreement, we compare the generated summaries directly by measuring similarity between the LLM-generated response (baseline) and our approach’s response on a per-video basis. We reported lexical overlap metrics (ROUGE-1/2/L F1 [2]) and semantic similarity metrics (BERTScore F1 [5] and embedding cosine similarity). ROUGE reflects n-gram overlap, whereas BERTScore and cosine similarity better capture paraphrasing and semantic equivalence.

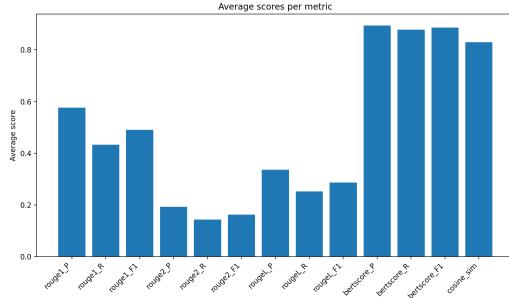


Figure 2: Average (across videos) comparison of generated summaries using ROUGE-1/2/L F1, BERTScore F1, and cosine similarity.

Results and trends. Figure 2 summarizes the *average* performance across all videos. This aggregate view provides the main conclusion of our response-level evaluation: our approach achieves consistently strong semantic agreement while maintaining competitive lexical overlap. Figure 3 shows per-video performance across all metrics. Overall, semantic metrics (BERTScore and cosine similarity) remain consistently high across videos, indicating that our summaries preserve the core meaning even when wording differs. In contrast, ROUGE scores exhibit larger variance across video IDs, which is expected because

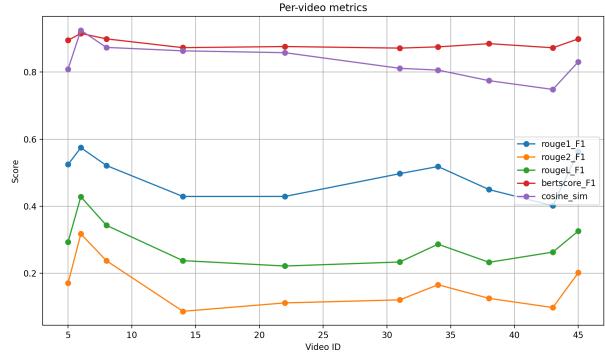


Figure 3: Per-video comparison of test-set video summaries using ROUGE-1/2/L F1, BERTScore F1, and cosine similarity.

small surface-form differences can significantly affect n-gram overlap.

Analysis. The gap between strong semantic similarity and more variable ROUGE suggests that our approach often produces semantically aligned but lexically different responses compared to the baseline. This is desirable in summarization settings where multiple valid phrasings exist. We also observe that ROUGE-2 is the most sensitive metric (largest fluctuations), reflecting its reliance on exact bigram matches; ROUGE-1 and ROUGE-L are comparatively more stable due to unigram and sequence-level matching. Finally, per-video variation highlights that summary difficulty is content-dependent, motivating the use of both lexical and semantic metrics for a balanced evaluation.

5.4 Error Analysis: Low ROUGE but Correct Semantics

Figure 4 illustrates a representative failure case where the ROUGE score is relatively low even though the generated summary preserves the main meaning of the original video. This happens because ROUGE is primarily an n-gram overlap metric, so paraphrasing and lexical variation (e.g., using different words for the same idea) can reduce the score despite semantic equivalence. In this example, the model output correctly captures the key semantics: the spring maintenance context, the focus on paper wasp nests commonly found under eaves, and the recommendation to remove or knock down nests early before colonies become established. However, the model abstracts and omits secondary details (e.g., that paper wasps can be beneficial insects, and specific guidance about

Original video

The video, presented by Jeff Edwards of the University of Wyoming Extension, discusses the Paper Wasp and how to manage its nests, particularly around homes and gardens. Edwards first points out that paper wasps are actually beneficial insects because they help control garden pests like caterpillars. However, their presence can be problematic because they are indiscriminate stingers when disturbed. Edwards recommends that when spring cleaning or gardening, homeowners should take the opportunity to remove any wasp nests found, especially those in the eaves of structures where they could pose a threat to people or pets. He demonstrates how a nest can be physically knocked down with a shovel or other implement and then thrown away. Alternatively, he suggests using chemical control specifically designed for wasp and hornet control. For chemical removal, he advises applying the product in the evening when all the wasps have returned to the nest and then cleaning the dead wasps and the nest off the structure.

Model output

The video provides a tip for homeowners during spring cleaning and gardening preparation, recommending the removal of certain insect nests found on homes and other structures. Specifically, it focuses on paper wasp nests, which are often found under the eaves. The speaker suggests that these nests, when discovered early, can be safely knocked down using a shovel or other implement. The key takeaway is to take the opportunity during spring maintenance to inspect and remove these nests, simplifying the process by addressing them while the colonies are still small or before they become fully established for the season.

Figure 4: Example where ROUGE underestimates summary quality: the model paraphrases the reference and omits minor details, but preserves the core semantics and key takeaway.

chemical control and timing), which further decreases lexical overlap and therefore ROUGE. This observation highlights the limitation of overlap-based metrics alone and motivates complementing ROUGE with semantic similarity measures (e.g., BERTScore/cosine similarity) and qualitative or human evaluation.

6 Conclusion

In this project, we successfully implemented the LLMVS framework for video summarization, demonstrating the effectiveness of leveraging Large Language Models for understanding and summarizing video content. Our approach achieves superior performance on the TVSum dataset compared to baseline methods, with Kendall’s τ of 0.177 and Spearman’s ρ of 0.229, substantially outperforming the A2Summ benchmark ($\tau=0.137$, $\rho=0.165$) and the zero-shot LLM baseline ($\tau=0.051$, $\rho=0.056$).

6.1 Key Findings

Effectiveness of LLM Embeddings: Our results validate that extracting and utilizing output embeddings from LLMs provides richer semantic information compared to using direct textual answers. This approach enables better capture of contextual relationships and narrative structure within videos.

Importance of Global Context: The significant performance improvement over the zero-shot LLM baseline highlights the critical role of the global context aggregator. The self-attention mechanism successfully integrates information across the entire

video sequence, enabling coherent and contextually aware summarization decisions.

Semantic Understanding: By centering the summarization process around textual descriptions and LLM reasoning rather than purely visual features, our framework demonstrates strong semantic understanding capabilities. The model effectively identifies action-oriented content and key narrative moments that align with human judgment.

Qualitative Analysis: Our generated summaries show good alignment with ground truth annotations, particularly in identifying important action sequences and minimizing redundant content. The model successfully captures high-importance segments such as demonstrations in how-to videos and dynamic actions in event footage.

6.2 Future Directions

While our results are promising, several areas warrant future exploration. First, extending evaluation to diverse datasets beyond TVSum (such as SumMe) would better establish generalization capabilities. Second, incorporating audio modality during summarization could enrich contextual understanding, particularly for videos with significant spoken content or music. Third, experimenting with different M-LLMs and LLMs could potentially improve both efficiency and effectiveness. Finally, investigating more sophisticated attention mechanisms or exploring larger window sizes might further enhance performance.

In conclusion, this project demonstrates that Large

Language Models, when properly integrated into video processing pipelines, can significantly advance video summarization capabilities by bringing semantic understanding and contextual reasoning to complement traditional visual analysis. The LLMVS framework represents a promising direction for future multimedia content analysis research, bridging the gap between visual data and language-based reasoning.

References

- [1] Min Jung Lee, Dayoung Gong, and Minsu Cho. Video summarization with large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18981–18991, 2025.
- [2] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- [3] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. Tvsum: Summarizing web videos using titles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5179–5187, 2015.
- [4] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [5] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020.