Too Many Frames, Not All Useful: Efficient Strategies for Long-Form Video QA

Jongwoo Park [1] **Kanchana Ranasinghe** [1] **Kumara Kahatapitiya** [1]
Wonjeong Ryu[2] **Donghyun Kim** [3] **Michael S. Ryoo** [1]

[1]Stony Brook University [2]KAIST AI [3]Korea University

jongwopark@cs.stonybrook.edu

Abstract

Long-form videos that span across wide temporal intervals are highly information redundant and contain multiple distinct events or entities that are often loosely related. Therefore, when performing long-form video question answering (LVQA), all information necessary to generate a correct response can often be contained within a small subset of frames. Recent literature leverage large language models (LLMs) in LVQA benchmarks, achieving exceptional performance, while relying on vision language models (VLMs) to convert all visual content within videos into natural language. Such VLMs often independently caption a large number of frames uniformly sampled from long videos, which is not efficient and can mostly be redundant. Motivated by this inefficiency, we propose LVNet, a modular and training-free framework featuring a novel Hierarchical Keyframe Selector (HKS) that efficiently selects a minimal set of informative frames tailored to each question. LVNet's modularity allows easy integration with existing approaches for more efficient LVQA. We achieve state-of-the-art performance among similarly configured models across four benchmark LVQA datasets: EgoSchema, NExT-QA, IntentQA, VideoMME. The code can be found at https://github.com/jongwoopark7978/LVNet

1 Introduction

Video understanding is a long-standing vision problem (Aggarwal and Ryoo, 2011) with numerous real-world applications. It has been traditionally studied even before the era of differentiable representation learning, with hierarchical approaches focusing on longer videos (Allen and Ferguson, 1994; Ivanov and Bobick, 2000; Shi et al., 2004; Hongeng et al., 2004; Ryoo and Aggarwal, 2006). Today, video understanding research involving the language modality is particularly popular, with tasks such as video question answering (QnA) that involve generating human-style conversations using large language models (LLMs) (Tapaswi et al., 2016; Zeng et al., 2017; Xu et al., 2017).

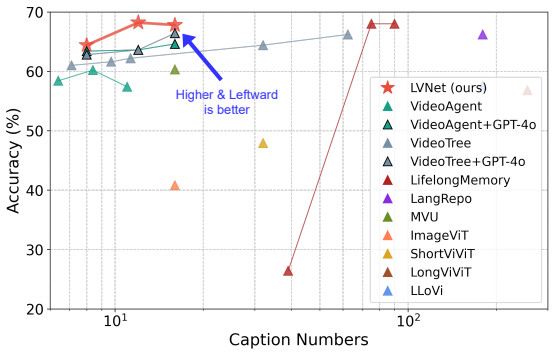

Figure 1: **High Accuracy with Low Compute:** LVNet achieves state-of-the-art performance on EgoSchema (subset) while processing only a fraction of frame captions (below 12 per video) with the expensive LLM. More detailed analysis presented in Section 4.4.

Motivated by successful image-language models (Liu et al., 2023; Dai et al., 2023), several works build video LLMs (Yu et al., 2023; Papalampidi et al., 2023; Maaz et al., 2023; Wang et al., 2024b) that directly process video frames in one stage to perform QnA. However, for long videos (containing 1000s of frames) these models require processing large visual token sequences with the LLM, making inference computationally expensive or even infeasible (see Table 5). An alternate line of works (Zhang et al., 2023; Wang et al., 2023; Ranasinghe et al., 2024) follow multi-stage pipelines that first extract frame-level information followed by temporal modeling with an LLM. While these multi-stage works similarly encounter compute bottlenecks with increased frame processing for longer videos, it is possible to feed the LLM with descriptors of non-uniformly sampled frames (Kahatapitiya et al., 2024; Wang et al., 2024d,f), motivating our exploration into *keyframe selection*, i.e. identifying a minimal set of frames most useful for correctly answering a given video-question pair.

Therein, we propose LVNet, a framework con-

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 3569–3588

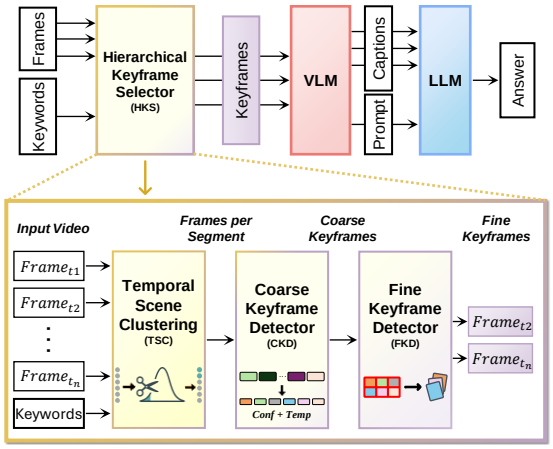

Figure 2: (top) **Overview:** LVNet uses a Hierarchical Keyframe Selector (HKS) module to select keyframes, followed by VLM & LLM for caption and answer generation. (below) **HKS Module** processes dense frames with lighter modules and progressively exploits heavier, more performance-oriented modules on smaller subsets of frames to ensure efficient computation.

taining a novel Hierarchical Keyframe Selector (HKS) that performs efficient key-frame selection followed by VLM for caption generation and LLM for answer generation as illustrated in Figure 2. Aligned with prior work (Zhang et al., 2023; Wang et al., 2024f,d), per-frame captions are processed with a powerful LLM to generate correct answers for a given video-question pair. As shown in Fig. 1, LVNet achieves strong performance efficiently, processing only a fraction of frame descriptors (captions) with the LLM. Compared to feeding all captions or all frames directly to a powerful model (e.g. GPT-4o), our LVNet performs inference at a fraction of the cost (see Table 6).

We summarize our key contributions as follows:

(a) **Efficient Frame Selection:** Hierarchical Keyframe Selector (HKS) efficiently extracts keyframes from sequences up to 1800 frames (hour long videos) with a filter rate over 98%.

(b) **Video Training Free:** Our framework requires no video level training and simply uses existing off-the-shelf modules.

(c) **Versatile Framework:** Existing methods can easily be integrated with LVNet to further boost their performance (details in Section A.7).

Proposed LVNet achieves state-of-the-art results (at common inference compute) on multiple long-form video question answering benchmarks demonstrating strong performance and generality.

2 Related Work

Video Question Answering: Visual question answering (VQA) involves generating open-ended

Feature	Ours	VA	Tr.	VT	VC	FV
(effective selection)						
Uses non-uniform sampling	✓	✓	✓	✓	✓	✓
Scene continuity-based selection	✓	✗	✗	✓	✓	✓
Robust to initial frames	✓	✗	✓	✓	✓	✓
Fine-grained visual refinement	✓	✗	✗	✗	✗	✗
(compute efficient)						
Lightweight feature extraction	✓	✗	✗	✗	✓	✓
Single pass inference	✓	✗	✗	✗	✓	✓
Video Training Free	✓	✓	✓	✗	✗	✗

Table 1: LVNet exhibits unique features compared to prior work VideoAgent (VA) (Fan et al., 2024), Traveler (Tr.) (Shang et al., 2024), VideoTree (VT) (Wang et al., 2024g), VideoChat-T (VC) (Zeng et al., 2024), and Frame-Voyager (FV) (Yu et al., 2024b) . See Section A.8 for details.

textual content conditioned on an image and natural language query (Agrawal et al., 2015). Its video variant, Video-VQA (Yu et al., 2019a) replaces images with videos. Multiple early datasets focus on querying objects or events based on referential and spatial relations (Xu et al., 2017; Zeng et al., 2017; Yu et al., 2019a). Later tasks require explicit temporal understanding of sequential events (Lei et al., 2018, 2020; Yu et al., 2019b). More recent datasets focus on longer videos containing multiple actions and scenes spread over wide time intervals (termed long-form videos) (Xiao et al., 2021; Li et al., 2022). Referred to as long-form video question answering (LVQA), these benchmarks are constructed to specifically test strong causal and temporal reasoning (Xiao et al., 2021) over long temporal windows (Mangalam et al., 2023). Some works tackling such video VQA tasks leverage graph networks to model cross object / event relations (Hosseini et al., 2022; Xiao et al., 2022a,b). A more recent line of works integrate LLMs to tackle this task (Zhang et al., 2023; Wang et al., 2023; Kahatapitiya et al., 2024; Wang et al., 2024d; Ranasinghe et al., 2024; Wang et al., 2024f; Fan et al., 2024) utilizing the strong reasoning skills of LLMs. A common aspect is the use of a vision language model (VLM) to convert frame level visual information into natural language. This in turn is input to the LLM which makes a final prediction.

Unlike these methods, LVNet incorporates a unique Hierarchical Keyframe Selector that progressively reduces the number of keyframe candidates. Lighter modules are applied to dense frames, while heavier, more performance-focused modules are applied to a small subset of filtered frames. Additionally, LVNet does not require video-level training, unlike earlier supervised approaches.

Frame Selection in Videos: The task of frame selection in videos has been long explored in video (Davis and Bobick, 1997; Zhao et al., 2017) with

more recent works focused directly on long-form video question answering (Buch et al., 2022; Wang et al., 2024g; Fan et al., 2024; Zeng et al., 2024; Yu et al., 2024b). Most similar to our work is Wang et al. (2024d) which employs an LLM based strategy for video frame selection. However, our LVNet differs with several unique features as summarized in Table 1.

3 Method

In this section, we present our training-free (*i.e.* zero-shot) framework for long-form video QA, LVNet. Videos are a dense form of data with even a few seconds long clip being composed of 100s of frames (individual images). In the case of long-form videos, this frame count is even greater. However, the information necessary to answer a given question is often contained in a handful of those frames. Our framework tackles this challenge of selecting an optimal and minimal set of informative frames. We refer to this as keyframe selection. Given such a set of useful frames, we also establish optimal strategies for extracting their information using modern large language models (LLMs), taking into account their sequential nature.

Our proposed LVNet comprises of three components: a Hierarchical Keyframe Selector (HKS), a Vision Language Model (VLM), and a Large Language Model (LLM) as illustrated in Figure 2. The HKS, an efficient, hierarchical keyframe selector, is the core contribution of our work. First, the model processes 900 uniformly sampled frames and clusters them into distinct scenes Next, it extracts keywords from a given natural language query via LLM and selects the frames most relevant to those keywords. Finally, the selected frames are described in natural language by a more powerful and computationally intensive VLM. Finally, an LLM processes the language descriptions of the selected frames to answer a given query.

3.1 Background

Recent approaches utilizing LLMs for long video question answering (LVQA) (Zhang et al., 2023; Wang et al., 2023; Kahatapitiya et al., 2024; Ranasinghe et al., 2024; Wang et al., 2024d) can be viewed as a composition of three sequential stages: a) frame selection, b) VLM based frame captioning, and c) LLM based answer generation. Note that the complexity of each stage varies across methods given their focus on different aspects of the LVQA task (*e.g.* frame selection in some is simply uniform sampling). In our work, we also follow this structure, but we focus on improving the frame selection stage. Under such a framework, our proposed HKS can serve as plug-in modules to replace the *frame selection* stage and the later two stages are similar to these prior works.

3.2 Architecture

Consider a video, $\mathbf{x} \in \mathbb{R}^{T\times C\times H\times W}$ with T, C, H, W for frames, channels, height, width respectively and its paired natural language query $\mathbf{q}$. Also consider a frame in $\mathbf{x}$ at timestamp t as $\mathbf{x}[t] \in \mathbb{R}^{C\times H\times W}$. Our goal is to output a response, referred as $\mathbf{r}$, suitable for the given query $\mathbf{q}$ based on information contained in the video $\mathbf{x}$.

Our LVNet processes a given video-query $(\mathbf{x}, \mathbf{q})$ pair to output a response, $\hat{\mathbf{r}}$. The HKS module initially processes this video-query pair, selects T' keyframes, and outputs a deterministically subsampled video $\mathbf{x}' \in \mathbb{R}^{T'\times C\times H\times W}$. Each of these T' frames is then passed through the captioning stage of our VLM to generate a set of natural language descriptions, $D = \{d_1, d_2, ...d_{T'}\}$ where d_i describes the frame $\mathbf{x}'[i]$. Finally, the LLM processes all descriptions D and the query $\mathbf{q}$ to generate response $\hat{\mathbf{r}}$. We illustrate this overall architecture in Figure 2.

3.3 Hierarchical Keyframe Selector

We now describe our proposed Hierarchical Keyframe Selector (HKS) module. As illustrated in Figure 2, our proposed HKS comprises of three sequential submodules, each reducing the frame count to T_a, T_b, and $T_c = T'$ respectively.

Temporal Scene Clustering (TSC): The role of TSC is to perform visual content aware preliminary frame sampling. The established approach for preliminary frame selection is uniform sampling (limited to at most 200 frames). In contrast, TSC processes 900 to 1800 uniformly sampled frames to extract per-frame visual features using a lightweight deep neural network (ResNet-18) followed by a clustering procedure to identify n non-overlapping frame sets. Within each of the n sets, we uniformly sample $\leq \tau$ frames obtaining a total of $T_a \leq \tau \times n$. Our iterative clustering procedure is outlined in Algorithm 1. It calculates pairwise distances between all frames accounting for intra-frame local information using the extracted per-frame features, followed by n iterative frame similarity based clustering operations. A single cluster could contain just one frame or significantly more based on frame feature similarities, leading to a *non-uniform sampling of frames* across the entire video. This allows

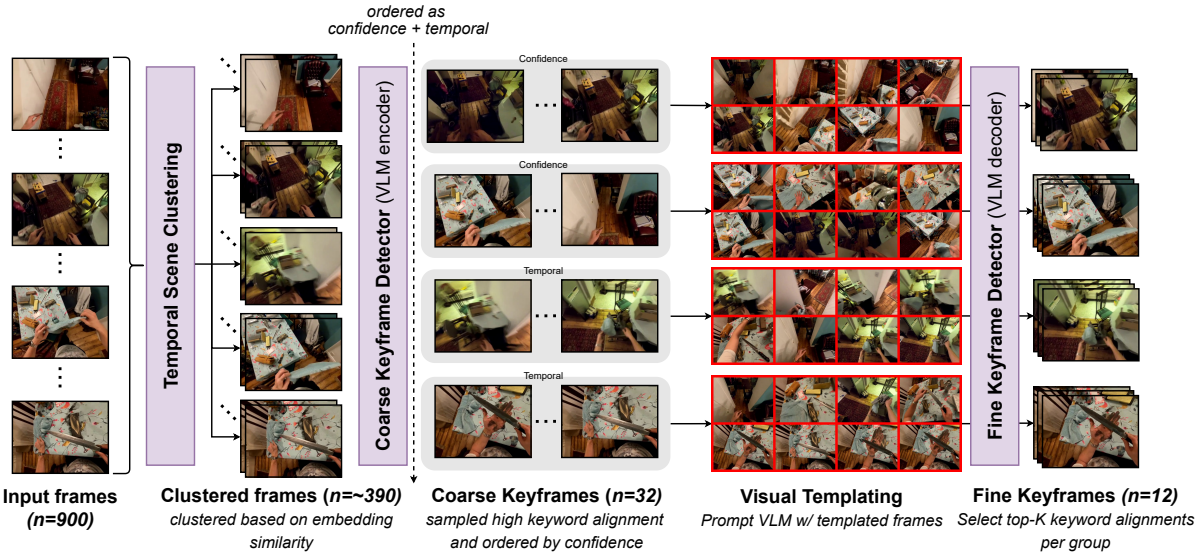

Figure 3: **Qualitative example**: We illustrate a challenging long-video QA scenario from EgoSchema (Mangalam et al., 2023). We consider an input of 900 frames, which first get clustered into scenes and subsampled to retain around 390 frames. Next, the Coarse Keyframe Detector selects only 32 frames out of them, based on the alignment with keywords (Here, keywords are extracted based on answer options, via an LLM). Such coarse keyframes are then ranked based on the combination of confidence value and temporal span, and grouped into four sets, each containing eight frames. These sets are then processed through visual templating (*i.e.* simple concatenation across space) and fed into a VLM for Fine Keyframe Detection, resulting in just 12 frames.

more frames to be sampled from the information heavy temporal regions of videos.

Coarse Keyframe Detector (CKD): Unlike TSC in the prior stage, CKD reasons across both visual and language modalities (using the paired textual query, q) to further sub-sample T_a into T_b frames. CKD contains three elements: keyword generation strategy, dual-encoder image-text model, and similarity based confidence assignment algorithm. Keyword generation utilizes the given query, $\mathbf{q}$, alongside hand-crafted templating operations or an LLM to select or generate suitable keywords. The dual-encoder image-text model uses a spatially aware contrastive language image pre-training (CLIP) network from (Ranasinghe et al., 2023). For confidence assignment, we construct an algorithm as outlined in Algorithm 2 which processes two lists, one of frames and one of keywords, and then calculates their pairwise likelihood of occurrence to assign each frame a confidence value (that reflects its usefulness to answer the query, $\mathbf{q}$). See Section A.5 for more details.

For a single query, there can be multiple regions in a video that are highly informative but not useful or relevant in answering that query. A single query can also contain multiple different concepts and attributes that must be given attention to construct a correct answer: the keyword generation attempts to capture each of these distinct attributes. On the visual modality, a single frame will also encode multiple concepts and attributes. Our design choice for the spatially aware CLIPpy dual-encoder VLM from (Ranasinghe et al., 2023) is motivated by this nature of individual frames. Finally, confidence assignment takes into account these multiple modes of information within each frame and the query to suitably assign confidence scores to each frame that reflects its query relevance. We also highlight how the confidence scores are directly linked to the related keyword (i.e. reason that makes the frame relevant), leading to better interpretability and the ability to perform further keyword-based refinement in later stages.

Fine Keyframe Detector (FKD): In the prior CKD stage, cross-modal selection utilizes a dual-encoder VLM that is constrained by the set of keywords provided and performs limited reasoning at frame level. In contrast, FKD uses a *visual templating module* to combine multiple frames and uses VLM to generate open-ended natural language output through higher-level reasoning. The input in this stage is the set of F_b frames, with each frame having an assigned confidence score and keyword.

Our visual templating module partitions the T_b frames into sets of 8 ordered by their confidence scores, arranges frame sets as grids to form a collage-style image, and annotates that image with visually identifiable tags corresponding to each

frame. We further illustrate this process in Figure 3 (see Visual Templating column). Each of these visual templated images also contain a subset of keywords that correspond to their 8 images. These resulting visual templated images along with a prompt containing their associated keywords and instructions to select a frame subset based on valid association between keywords and images (see Section A.6 for details) are input to the VLM. The output of the VLM is used to select a subset of each 8 image group. These frames are collected as output of the FKD stage, overall resulting in T_c frames.

The purpose of the initial visual templating module is to allow reasoning across a set of frames using the image-text VLM (which is trained to process a single image at time). This partitioning of the input T_b frames is performed based on confidence scores from the prior stage and timestamps. The eight frames with top confidence scores are grouped into the first visual template, followed by the next eight and so forth. This ensures the VLM selects both high confidence concepts and low confidence concepts, accounting for biases and weaknesses in our CKD stage. After that, we temporally reorder some image sets with low confidence scores to cover keyframes distributed across long-range segments, while the sets with high confidence scores concentrate on keyframes in short-range segments. A total of 16 low-score frames are temporally reordered in this process. The algorithm is described in Algorithm 3 and the prompting technique is explained in Section A.6. Our intuition is that such a mechanism allows one to best utilize the complementary strengths of two different VLMs from CKD and FKD stages for better frame selection overall.

4 Experiments

In this section, we first discuss our experimental setup followed by quantitative evaluations comparing to existing baselines and ablations of our proposed components. We then discuss the results on efficiency and scalability, and finally outline limitations.

4.1 Experimental Setup

Datasets: Given the training free nature of our framework, we do not utilize any video datasets for training. Datasets are used purely for evaluation. We select three benchmark video visual question answering datasets focused on long-form videos for this purpose: EgoSchema (Mangalam et al., 2023), NExT-QA (Xiao et al., 2021), and IntentQA (Li et al., 2023). In addition, to further highlight the strength of our approach on longer videos, we include results on VideoMME's long split (Fu et al., 2024). These datasets are public available and can be used freely for academic research. The first dataset, EgoSchema, consists of 5031 questions and each video lasts three-minute and have multiple choice question. The second dataset, NExT-QA, is another rigorously designed video question answering benchmark containing questions that require causal & temporal action reasoning, and common scene comprehension to correctly answer. These questions are further classified as Causal (Cau.), Temporal (Tem.), and Descriptive (Des.) and we evaluate on its validation set containing 4996 questions over 570 videos. The third dataset, IntentQA, is based on NExT-QA videos corresponding to temporal and causal reasoning quetions. It consists of 16k multiple-choice questions which are classified as Why?, How? or Before/After (B./A.). The fourth dataset, VideoMME, consists of very long videos—some up to one hour long, with an average duration of 44 minutes, and provides 900 Q&A. Collectively, these benchmarks span **three orders of magnitude in duration**—from short clips ($\sim$44 s on NExT-QA/IntentQA), to minute-scale videos (EgoSchema, $\approx$3 min), to hour-scale footage (VideoMME's long split, 30–60 min)—allowing us to stress-test LVNet *from seconds to hour-long videos* under a single, training-free evaluation protocol.

Model Choices & Hyperparameters: For the HKS module, we use the ResNet-18 (He et al., 2016a) for the TSC, CLIP-B/16 (Ranasinghe et al., 2023) for the CKD and GPT-4o for the FKD. We select ResNet-18 and CLIP-B/16 due to their smaller models sizes—0.01B and 0.12B, respectively–which are significantly lighter compared to LLMs that are on a billion parameter scale. In line with previous state-of-the-art work (Wang et al., 2024f; Zhang et al., 2023; Wang et al., 2023), we use up to 4 NVIDIA RTX A5000 GPUs and GPT or DeepSeek APIs for running baselines and our setup in all evaluations. Also following prior work, we report results over single evaluation runs.

4.2 Evaluation

Quantitative Results: We evaluate LVNet on the EgoSchema, NExT-QA, and IntentQA dataset and present our results in Table 2. Models with video-caption pretraining are de-emphasized in grey to ensure fairness with image-level pertaining. Mod-

Model	EgoSchema		NExT-QA		IntentQA		VT Free
	Cap.↓	Acc.↑ (%)	Cap.↓	Acc.↑ (%)	Cap.↓	Acc.↑ (%)	
VideoLLaMA 2 (Cheng et al., 2024)	-	53.3	-	-	-	-	no
InternVideo2 (Wang et al., 2024e)	-	60.2	-	-	-	-	no
Tarsier (Wang et al., 2024a)	-	61.7	-	79.2	-	-	no
Vamos (Wang et al., 2023)	-	48.3	-	-	-	-	yes
IG-VLM (Kim et al., 2024)	-	59.8	-	68.6	-	65.3	yes
VIOLET (Fu et al., 2023)	5	19.9	-	-	-	-	yes
mPLUG-Owl (Ye et al., 2023)	5	31.1	-	-	-	-	yes
VideoAgent (Wang et al., 2024d)	8.4	54.1	8.2	71.3	-	-	yes
MVU (Ranasinghe et al., 2024)	16	37.6	16	55.2	-	-	yes
MoReVQA (Min et al., 2024)	30	51.7	16	69.2	-	-	yes
VFC (Momeni et al., 2023)	-	-	32	51.5	-	-	yes
ProViQ (Choudhury et al., 2023)	50	57.1	50	64.6	-	-	yes
VideoTree (Wang et al., 2024g)	62.4	61.1	(56)	73.5	(56)	66.9	yes
FrozenBiLM (Yang et al., 2022)	90	26.9	-	-	-	-	yes
LifelongMemory (Wang et al., 2024f)	90	62.1	-	-	-	-	yes
TraveLER (Shang et al., 2024)	(101)	53.3	(65)	68.2	-	-	yes
LangRepo (Kahatapitiya et al., 2024)	180	41.2	90	60.9	90	59.1	yes
LLoVi (Zhang et al., 2023)	180	50.3	90	67.7	90	64.0	yes
LVNet (ours)	12	61.1	12	72.9	12	71.7	yes

Table 2: **Long Video Evalation:** LVNet achieves state-of-the-art accuracies of 61.1%, 72.9%, and 71.7% on EgoSchema, NExT-QA, and IntentQA datasets respectively using just 12 frames compared to models using a similar number of captions. Models are ordered based on number of captions processed per video. Models with video-level training (VT) or utilizing significantly more captions than 12 frames used by LVNet are de-emphasized in grey or downplayed in light green to ensure fair comparison. Numbers in parentheses () indicate the maximum number of frames used. See Sec. A.2 in appendix for detailed results.

Method	**LLM Param/Type**	**VT Free**	**TS Cap.↓**	**Acc.↑**
VideoChat-T	7B/OS	no	N/A	43.8
Frame-Voyager	34B/OS	no	N/A	51.2
LVNet (DS-V3)	37B/OS	yes	24	**53.1**
VideoAgent+GPT-4o	<1.8T/PP	yes	24	51.3
VideoTree+GPT-4o	<1.8T/PP	yes	24	52.3
LVNet (GPT-4o)	<1.8T/PP	yes	24	**53.9**

Table 3: **Comparison to Keyframe Selection Methods on VideoMME:** We compare LVNet with both single-stage methods that rely on video-level training (VideoChat-T, Frame-Voyager) and two-stage methods (VideoAgent, VideoTree). For the two-stage comparison, we evaluate VideoTree and VideoAgent with GPT-4o as both the captioner and LLM under the same 24-frame caption budget for a direct head-to-head comparison. *Legend: DS-V3=DeepSeek-V3, OS=open-source, PP=proprietary, VT Free=no video-level training, TS Cap.=two-stage caption numbers.*. Notably, LVNet outperforms both single-stage and two-stage methods without requiring video-level training.

els utilizing significantly more captions than the 12 frames are downplayed in light green to consider caption efficiency. We reiterate how number of captions input to LLM affects inference compute cost of methods the most.

For EgoSchema(fullset), LVNet achieves 61.1%, the highest among the models utilizing approximately 12 captions. This result outperforms VideoAgent, the next best model using 8.4 captions, by +7%, is on par with VideoTree while using only 1/5 of the captions, and outperforms TraveLER by +7.8% while utilizing only 12% of the captions.

We next evaluate on NExT-QA dataset, which has a particular focus on both temporal and causal reasoning based question-answer pairs. LVNet achieves state-of-the-art performance on this benchmark outperforming prior work among the models utilizing approximately 12 captions. In fact, our LVNet outperforms VideoAgent by +1.6%.

In IntentQA dataset, LVNet outperforms all prior work, including de-emphasized models with video-caption pretraining and downplayed models utilizing significantly compute (captions input to LLM) than ours. In fact, LVNet shows a substantial improvement of +4.8% over the next best VideoTree, while using only 13% of captions (12 vs 90).

Lastly, Table 3 shows LVNet's performance on VideoMME's long split (Fu et al., 2024), which features videos up to an hour long—significantly exceeding the 12-minute average in MLVU (Zhou et al., 2024) or the 17-minute average in the overall VideoMME. In combination with an open-source LLM (DeepSeek-V3), LVNet outperforms the single-stage keyframe selection methods VideoChat-T (Zeng et al., 2024) and Frame-Voyager (Yu et al., 2024b) by +9.3% and +1.9%, respectively, without any video-level training. Moreover, under a direct head-to-head setting where VideoAgent, VideoTree, and LVNet all use GPT-4o as both the frame captioner and LLM with the same 24-frame budget, LVNet achieves 53.9% accuracy, outperforming VideoAgent(51.3%) and

Model	Frame Captions		
	8	12	16
VideoAgent+GPT-4o	63.4	63.6	64.6
VideoTree+GPT-4o	62.8	63.6	66.4
LVNet (GPT-4o)	**64.4**	**68.2**	**67.8**

(a) **Frame Caption Budget**: LVNet ourperforms VideoAgent and VideoTree at all caption budgets when all methods use GPT-4o as both the frame captioner and LLM.

Templating Order	Acc. ↑
Temporal	65.2
Confidence	67.6
Hybrid (both)	**68.2**

(b) **Visual Templating**: Combination of confidence-based & temporal ordering gives the best performance.

TSC	CKD	FKD	Acc. ↑
✗	✗	✗	62.6
✓	✗	✗	64.5
✓	✓	✗	65.8
✓	✓	✓	**68.2**

(c) **HKS Ablation**: LVNet accuracy consistently improves with each HKS sub-module.

Table 4: **Ablation study** on EgoSchema (Mangalam et al., 2023): We evaluate different design decisions of our framework on EgoSchema 500-video subset for zero-shot VQA.

Method	LLM AP	Frames ↑	LLM frames ↓	Acc. ↑
Qwen-VL	7B	128	128	37.8
Qwen-VL	7B	256	256	OOM
Qwen-VL	7B	1800	1800	OOM
LVNet (DS-V3)	37B	1800	24	53.1

Table 5: **Single-Stage Method Comparison:** We report Accuracy on VideoMME long split (average video length 41 mins) along with LLM active parameters (LLM AP), total frames processed, and frames input to LLM as tokens / captions (LLM frames). LVNet DeepSeek-V3 (DS-V3) variant is used. Processing lengthy frame sequences with single-stage models at fixed compute becomes infeasible. Inference tested on 4 x 24GB A5000 GPUs. Similar findings in Zeng et al. (2024).

Model	FV ↑	FL ↓	VC ↓	TC ↓	Acc. ↑
GPT-4o	384	384	$2.88	∼$2592	65.3
LVNet (GPT-4o)	1800	24	$0.19	$171	53.9

Table 6: **API Comparison:** We perform cost comparison with using GPT-4o directly on frames vs with LVNet. We report accuracy on VideoMME long split (Acc), frames processed per video (FV), frames / captions input to LLM (FL), per video cost (VC), and total evaluation cost (TC). Our LVNet achieves competitive performance at over 10x less inference cost. GPT-4o accuracy result from official benchmark leaderboard.

VideoTree(52.3%) by +2.6% and +1.6%, respectively, further confirming its effectiveness on very long videos. See Section A.3 for the detailed results.

Qualitative Examples: Due to space, qualitative open-ended responses comparing LVNet to uniform sampling are provided in the Appendix (see Figure A.5).

4.3 Efficiency & Scalability

Stage-wise runtime, FPS, and memory. We report a detailed efficiency profile for each LVNet component on a single VideoMME-long video (RTX-6000 Ada). As shown in Table 7, the three-stage HKS (TSC→CKD→FKD) contributes 36% of the total runtime, confirming that most compute (64%) lies in captioning and LLM Q&A, which are common overheads to two-stage pipelines.

Stages	FPS	Mem.	Frames	Lat. (s)	Lat. (%)
HKS-1/3 (TSC)	182	19	1800	9.9	16
HKS-2/3 (CKD)	138	9	427	3.1	5
HKS-3/3 (FKD)	1	-	8	9.2	15
Captioning	1	-	24	38.7	62
LLM Q&A	-	-	-	1.0	2

Table 7: **LVNet Stage-wise Performance:** The table reports per-module elapsed time, FPS, and peak GPU memory for a single VideoMME-long video processed on an RTX-6000 Ada. Within LVNet, the three stage Hierarchical Keyframe Selectors contributes 36 % of the total runtime. It indicates that most compute lies in the captioning and LLM Q&A steps, which are overheads shared by all two-stage pipelines, rather than in the keyframe selection. Mem. and Lat. in the header stands for the GPU memory (GB) and Latency.

Direct runtime comparison to two-stage baselines. Complementing the internal latency breakdown in Table 7, we measure end-to-end per-video inference time for LVNet, VideoAgent, and VideoTree under matched conditions on a single NVIDIA RTX A5000 GPU with a 180-frame input video. As shown in Table 8, LVNet finishes in 25.6 s (7.0 FPS), achieving 3.4× and 2.5× speedups over VideoAgent (87.3 s, 2.1 FPS) and VideoTree (64.2 s, 2.8 FPS), respectively. The main gap comes from early-stage iterations: VideoAgent takes 40.5 s in the initial selection stage as it invokes a heavy LLM iteratively in every step when constructing keyframe grids. VideoTree takes 52.4 s as it similarly runs a heavy VLM captioner and LLM repeatedly in the first stage (Adaptive Breadth Expansion) to form keyframe clusters. This early-stage, iterative use

Model	Inference Time (s)↓	FPS↑
VideoAgent	87.3	2.1
VideoTree	64.2	2.8
LVNet	**25.6**	**7.0**

Table 8: **Direct runtime comparison to two-stage baselines.** Wall-clock inference time on a 180-frame input video measured using a single NVIDIA RTX A5000 GPU. LVNet achieves 3.4× and 2.5× speedups over VideoAgent and VideoTree, respectively.

Model	Frames	Memory (GB)	Inference time (s)	FPS
Qwen2.5-VL-72B	8	355	11.8	0.7
Qwen2.5-VL-72B	12	373	16.8	0.7
Qwen2.5-VL-72B	24	OOM	OOM	OOM
LVNet	1800	19	61.9	29.1

Table 9: **Model Efficiency Comparison:** End-to-end, LVNet handles 1,800 frames in 61.9 s (29.1 FPS) while using just 19 GB on a single GPU, whereas Qwen2.5-VL-72B requires eight GPUs and 373 GB yet runs $\approx 40\times$ slower at 0.7 FPS. Despite architectural and hardware differences, these figures show that LVNet keyframe selector is lightweight and that the overall system delivers high FPS while remaining single-GPU friendly for long-form video.

of heavy VLM/LLM modules dominates their latency, and many of the frames they process turn out to be uninformative for answering the question. On the other hand, LVNet's lightweight HKS (TSC+CKD+FKD) takes only 6.7 s and applies the expensive captioner+LLM only once on the final selected frames.

End-to-end throughput and single-GPU footprint. Table 9 contrasts end-to-end wall-clock speed and memory. LVNet processes 1,800 frames in 61.9 s (29.1 FPS) on a single GPU using 19 GB, while Qwen2.5-VL-72B requires 8 GPUs and 373 GB yet runs ∼40× slower at 0.7 FPS. Qwen2.5-VL-72B encounters out of memory (OOM) at 24 frames. Despite architectural and hardware differences, Table 7 and Table 9 show that LVNet's keyframe selector is scalable to process large number of frames in long-form video as it is lightweight, single-GPU-friendly, and provides high FPS.

Single-stage scaling limits on long videos. Table 5 shows that single-stage models (e.g., Qwen-VL (Bai et al., 2023)) run OOM beyond 128 frames, whereas LVNet achieves higher accuracy with only 24 keyframes passed to the LLM, maintaining tractable compute. It shows the LVNet's scalability of processing large number of frames with high accuracy.

API cost comparison. Table 6 compares inference cost and performance against purely GPT-4o prompting, showing that LVNet delivers competitive accuracy with over 10× lower per-video LLM cost, thanks to efficient keyframe selection.

Frames	Acc. (%)
12	50.1
24	**53.9 (+3.8)**

(a) **# Frames in VideoMME.**

LLM	Acc. (%)
GPT-3.5	61.0
GPT-4	65.4
GPT-4o	**68.2**

(b) **Choice of LLM**

Table 10: **Scalability of LVNet.** Left: accuracy improves as the frame budget increases for very long videos. Right: newer LLMs yield higher accuracy without retraining, reflecting LVNet's modularity.

Scalability across frame budget and LLM upgrades. Table 10 summarizes two complementary scaling axes. *Left:* on VideoMME-Long, increasing the frame budget from 12 to 24 yields a +3.8% gain. Conversely, on EgoSchema's 3-min videos, the 12-frame setting is optimal (68.2 %), indicating extra keyframes help LVNet for longer videos. It is consistent with our keyframe-driven design that benefits from additional informative frames when videos are very long. *Right:* LVNet improves monotonically with stronger LLMs (GPT-3.5 → GPT-4 → GPT-4o) without any retraining, highlighting the modularity and scalability of LVNet with newer LLMs.

4.4 Ablations

In this section, we present ablations on key design decisions such as the sorting order in FKD, the number of frames for captions, and the effect of different components in HKS. In all ablations, we use a subset of EgoSchema (Mangalam et al., 2023), composed of 500 videos. Additional ablations about *Choice of LLM* and *Effect of Patch Size on Keyword Matching in CKD* are in Section A.1

Number of Frame Captions: We conduct an ablation on the number of frame captions, directly comparing our approach with VideoAgent (Wang et al., 2024d) and VideoTree (Wang et al., 2024g) under the same GPT-4o backbone for both frame captioning and LLM reasoning. As shown in Table 4a, LVNet achieves the highest accuracy of 68.2% with 12 captions, outperforming both VideoAgent and VideoTree (63.6%) by +4.6% absolute. LVNet also consistently outperforms the baselines at 8 captions (+1.0% / +1.6%) and 16 captions (+3.2% / +1.4%) compared to VideoAgent / VideoTree..

Visual Templating Order: In visual templating, prioritizing frames by keyword confidence scores followed by reordering low-confidence frames based on timestamp proves more effective than using confidence scores or temporal order alone, as shown in Table 4b. In this hybrid approach, high-confidence frames capture short but important segments of videos, while low-confidence keyframes, which are crucial but visually challenging for keyword matching, are temporally ordered to cover broader segments. This hybrid approach outperforms solely temporal ordering or confidence-based ordering by +3% and +0.6%, respectively.

Effect of Hierarchical Keyframe Modules: Table 4c demonstrates the impact of incrementally adding the temporal scene clustering (TSC), coarse keyframe detector (CKD), and fine keyframe detector (FKD) modules. Without any of these modules, the model relies on uniform sampling and achieves 62.6%. When TSC is added and 12 frames are selected uniformly, the accuracy increases to 64.5%. Adding both TSC and CKD raises the accuracy to 65.8%. Finally, incorporating all three modules—TSC, CKD, and FKD—into the model, which is LVNet, results in an accuracy of 68.2%. This demonstrates the importance of including all modules in LVNet for optimal performance.

5 Conclusion

We proposed a novel approach for Long-form Video Question Answering (LVQA) that achieves state-of-the-art performance compared to the model using the similar-scale captions across four benchmarks. Our Hierarchical Keyframe Selector demonstrates the effectiveness of keyframe selection in understanding a very long-form video QA. Additionally, we highlight the zero-shot capability for long-form video comprehension of our LVNet framework, which requires no video-level training. Our experiments showcase its significant advantage over existing single-stage and two-stage keyframe selectors.

Limitations

Despite the effectiveness of LVNet, as demonstrated by benchmark experiments and comprehensive ablations, our study has certain limitations, which we discuss below.

- First, we acknowledge that we are unable to evaluate LVNet and other models with all available VLMs or LLMs due to computational constraints and high costs. However, we carefully select GPT-4o and DeepSeek-v3, the LLMs commonly used in video understanding research, for our main experiments and provide ablation studies comparing various LLMs (*e.g.* GPT-3.5, GPT-4, and GPT-4o) to ensure a fair performance comparison, as presented in Table 4a and Table 10b.
- Our hierarchical keyframe selector consists of three components: TSC, CKD, and FKD. While we demonstrated the effectiveness of each component in Table 4c, we did not have the time or resources to develop a unified module that could replace all three. Although this is beyond the scope of this paper, exploring a more efficient implementation that integrates these three modules into a single model would be an interesting direction for future research.
- Like any LLM-based approach, LVNet is sensitive to prompting. To ensure the transparency, we provide examples of these prompts in Figure A.5 and Figure A.6. We also plan to release the code to enable further exploration by other researchers.
- Finally, we acknowledge that, as our approach is zero-shot, any inherent limitations or biases in the pretrained models may persist in the outputs of LVNet.

Acknowledgements: This research was financially supported by the Ministry of Trade, Industry, and Energy (MOTIE), Korea, under the "Global Industrial Technology Cooperation Center(GITCC) program" supervised by the Korea Institute for Advancement of Technology (KIAT).(Task No. P0028420). This research was supported by the National Research Council of Science & Technology(NST) grant by the Korea government(MSIT) (No. GTL25041-000).

References

Jake K. Aggarwal and Michael S. Ryoo. 2011. Human activity analysis. *ACM Computing Surveys (CSUR)*, 43:1 – 43.

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Devi Parikh, and Dhruv Batra. 2015. Vqa: Visual question answering. *International Journal of Computer Vision*, 123:4 – 31.

James F Allen and George Ferguson. 1994. Actions and events in interval temporal logic. *Journal of logic and computation*, 4(5):531–579.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *Preprint*, arXiv:2308.12966.

S. Buch, Cristobal Eyzaguirre, Adrien Gaidon, Jiajun Wu, Li Fei-Fei, and Juan Carlos Niebles. 2022. Revisiting the "video" in video-language understanding. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2907–2917.

Yukang Chen, Fuzhao Xue, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, Ethan He, Hongxu Yin, Pavlo Molchanov, Jan Kautz, Linxi Fan, Yuke Zhu, Yao Lu, and Song Han. 2024a. Longvila: Scaling long-context visual language models for long videos. *Preprint*, arXiv:2408.10188.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. 2024b. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. 2024c. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 67(12):220101.

Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. 2024. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*.

Rohan Choudhury, Koichiro Niinuma, Kris M Kitani, and László A Jeni. 2023. Zero-shot video question answering with procedural programs. *arXiv preprint arXiv:2312.00937*.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. *arXiv preprint arXiv:2305.06500*.

James Davis and Aaron Bobick. 1997. The representation and recognition of action using temporal templates. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744.

Yue Fan, Xiaojian Ma, Rujie Wu, Yuntao Du, Jiaqi Li, Zhi Gao, and Qing Li. 2024. Videoagent: A memory-augmented multimodal agent for video understanding. *arXiv preprint arXiv:2403.11481*.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. 2024. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*.

Tsu-Jui Fu, Linjie Li, Zhe Gan, Kevin Lin, William Yang Wang, Lijuan Wang, and Zicheng Liu. 2023. An empirical study of end-to-end video-language transformers with masked visual modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22898–22909.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016b. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Somboon Hongeng, Ram Nevatia, and Francois Bremond. 2004. Video-based event recognition: activity representation and probabilistic recognition methods. *Computer Vision and Image Understanding*, 96(2):129–162.

Pedram Hosseini, David A. Broniatowski, and Mona Diab. 2022. Knowledge-augmented language models for cause-effect relation classification. In *Proceedings of the First Workshop on Commonsense Representation and Reasoning (CSRR 2022)*, pages 43–48, Dublin, Ireland. Association for Computational Linguistics.

Yuri A. Ivanov and Aaron F. Bobick. 2000. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):852–872.

Kumara Kahatapitiya, Kanchana Ranasinghe, Jongwoo Park, and Michael S Ryoo. 2024. Language repository for long video understanding.

Wonkyun Kim, Changin Choi, Wonseok Lee, and Wonjong Rhee. 2024. An image grid can be worth a video: Zero-shot video question answering using a vlm. *arXiv preprint arXiv:2403.18406*.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara Berg. 2018. TVQA: Localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jie Lei, Licheng Yu, Tamara Berg, and Mohit Bansal. 2020. TVQA+: Spatio-temporal grounding for video question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8211–8225, Online. Association for Computational Linguistics.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024. Llava-onevision: Easy visual task transfer. *Preprint*, arXiv:2408.03326.

Jiangtong Li, Li Niu, and Liqing Zhang. 2022. From representation to reasoning: Towards both evidence and commonsense reasoning for video question-answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jiapeng Li, Ping Wei, Wenjuan Han, and Lifeng Fan. 2023. Intentqa: Context-aware video intent reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11963–11974.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

Muhammad Maaz, Hanoona Abdul Rasheed, Salman H. Khan, and Fahad Shahbaz Khan. 2023. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *Annual Meeting of the Association for Computational Linguistics*.

Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. 2023. Egoschema: A diagnostic benchmark for very long-form video language understanding. *ArXiv*, abs/2308.09126.

Juhong Min, Shyamal Buch, Arsha Nagrani, Minsu Cho, and Cordelia Schmid. 2024. Morevqa: Exploring modular reasoning models for video question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13235–13245.

Liliane Momeni, Mathilde Caron, Arsha Nagrani, Andrew Zisserman, and Cordelia Schmid. 2023. Verbs in action: Improving verb understanding in video-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15579–15591.

Pinelopi Papalampidi, Skanda Koppula, Shreya Pathak, Justin Chiu, Joe Heyward, Viorica Patraucean, Jiajun Shen, Antoine Miech, Andrew Zisserman, and Aida Nematzdeh. 2023. A simple recipe for contrastively pre-training video-first encoders beyond 16 frames. *arXiv preprint arXiv:2312.07395*.

Kanchana Ranasinghe, Xiang Li, Kumara Kahatapitiya, and Michael Ryoo. 2024. Understanding long videos in one multimodal language model pass.

Kanchana Ranasinghe, Brandon McKinzie, Sachin Ravi, Yinfei Yang, Alexander Toshev, and Jonathon Shlens. 2023. Perceptual grouping in contrastive vision-language models. In *ICCV*.

Michael S. Ryoo and Jake K. Aggarwal. 2006. Recognition of composite human activities through context-free grammar based representation. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2:1709–1718.

Chuyi Shang, Amos You, Sanjay Subramanian, Trevor Darrell, and Roei Herzig. 2024. Traveler: A modular multi-lmm agent framework for video question-answering. *arXiv preprint arXiv:2404.01476*.

Yifan Shi, Yan Huang, David Minnen, Aaron Bobick, and Irfan Essa. 2004. Propagation networks for recognition of partially ordered sequential action. In *CVPR*.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. MovieQA: Understanding stories in movies through question-answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jiawei Wang, Liping Yuan, and Yuchen Zhang. 2024a. Tarsier: Recipes for training and evaluating large video description models. *arXiv preprint arXiv:2407.00634*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Ke-Yang Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *ArXiv*, abs/2409.12191.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024c. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Shijie Wang, Qi Zhao, Minh Quan Do, Nakul Agarwal, Kwonjoon Lee, and Chen Sun. 2023. Vamos: Versatile action models for video understanding. *arXiv preprint arXiv:2311.13627*.

Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. 2024d. Videoagent: Long-form video understanding with large language model as agent.

Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Jilan Xu, Zun Wang, et al. 2024e. Internvideo2: Scaling video foundation models for multimodal video understanding. *arXiv preprint arXiv:2403.15377*.

Ying Wang, Yanlai Yang, and Mengye Ren. 2024f. Lifelongmemory: Leveraging llms for answering queries in long-form egocentric videos. *Preprint*, arXiv:2312.05269.

Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. 2024g. Videotree: Adaptive tree-based video representation for llm reasoning on long videos. *arXiv preprint arXiv:2405.19209*.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. 2021. NExT-QA: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Junbin Xiao, Angela Yao, Zhiyuan Liu, Yicong Li, Wei Ji, and Tat-Seng Chua. 2022a. Video as conditional graph hierarchy for multi-granular question answering. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, pages 2804–2812.

Junbin Xiao, Pan Zhou, Tat-Seng Chua, and Shuicheng Yan. 2022b. Video graph transformer for video question answering. In *European Conference on Computer Vision*, pages 39–58. Springer.

Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. 2017. Video question answering via gradually refined attention over appearance and motion. In *ACM Multimedia*.

Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. 2022. Zero-shot video question answering via frozen bidirectional language models. *Advances in Neural Information Processing Systems*, 35:124–141.

Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. 2023. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*.

Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. 2023. Self-chained image-language model for video localization and question answering. *ArXiv*, abs/2305.06988.

Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. 2024a. Self-chained image-language model for video localization and question answering. *Advances in Neural Information Processing Systems*, 36.

Sicheng Yu, Chengkai Jin, Huanyu Wang, Zhenghao Chen, Sheng Jin, Zhongrong Zuo, Xiaolei Xu, Zhenbang Sun, Bingni Zhang, Jiawei Wu, et al. 2024b. Frame-voyager: Learning to query frames for video large language models. *arXiv preprint arXiv:2410.03226*.

Zhou Yu, D. Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. 2019a. Activitynet-qa: A dataset for understanding complex web videos via question answering. *ArXiv*, abs/1906.02467.

Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. 2019b. ActivityNet-QA: A dataset for understanding complex web videos via question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Kuo-Hao Zeng, Tseng-Hung Chen, Ching-Yao Chuang, Yuan-Hong Liao, Juan Carlos Niebles, and Min Sun. 2017. Leveraging video descriptions to learn video question answering. In *AAAI Conference on Artificial Intelligence*.

Xiangyu Zeng, Kunchang Li, Chenting Wang, Xinhao Li, Tianxiang Jiang, Ziang Yan, Songze Li, Yansong Shi, Zhengrong Yue, Yi Wang, et al. 2024. Timesuite: Improving mllms for long video understanding via grounded tuning. *arXiv preprint arXiv:2410.19702*.

Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. 2023. A simple llm framework for long-range video question-answering. *arXiv preprint arXiv:2312.17235*.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. 2024a. Video instruction tuning with synthetic data. *Preprint*, arXiv:2410.02713.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. 2024b. Video instruction tuning with synthetic data. *Preprint*, arXiv:2410.02713.

Zhichen Zhao, Huimin Ma, and Shaodi You. 2017. Single image action recognition using semantic body part actions. In *The IEEE International Conference on Computer Vision (ICCV)*.

Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. 2024. Mlvu: A comprehensive benchmark for multi-task long video understanding. *arXiv preprint arXiv:2406.04264*.

Appendix

A.1 Additional Ablations

In this section, we present additional experiments conducted to inform the LVNet's design. We have tested different LLMs and experimented with various scales of the visual feature map.

Model	Frames	Acc. (%)
GPT-4o	12	62.6
LVNet	8	64.4
LVNet	12	68.2

(a) **Uniform vs LVNet**.

Patch Size	Acc. (%)
1x1	63.6
7x7	66.2
14x14	68.2

(b) **Effect of Patch Size in CKD**: A larger patch size in Keyword Matching performs better.

Table A.11: **Additional comparisons** on EgoSchema subset. Left: uniform 12-frame sampling (GPT-4o) vs LVNet at 8 and 12 frames. Right: effect of patch size in CKD.

Uniform Sampling Comparison: GPT-4o with uniform 12-frame sampling on the EgoSchema subset scores 62.6%. LVNet outperforms GPT-4o, reaching 64.4% with just 8 frames and 68.2% with 12 frames (Table A.11a). To keep the evaluation both methodical and cost-effective, we first ran GPT-4o and LVNet on the EgoSchema subset as validation to find the optimal settings, then transferred those settings to the full EgoSchema test set and the remaining benchmarks. This procedure avoids the $>$ \$3,000 API expense of running GPT-4o uniformly across all four datasets while still demonstrating LVNet's stronger generalization. For LLM choices, see Sec. 4.3.

Effect of Patch Size on Keyword Matching in CKD: Table A.11b shows the effect of the scales of the patch sizes in the CKD. Since keywords can represent activities spanning the entire image or confined to a small region, we adjust the resolution of the visual feature map output from the spatially aware contrastive image pre-training (CLIP) network (Ranasinghe et al., 2023) to match keywords. Our findings show that higher resolutions lead to better accuracy. In LVNet, we use a 14×14 feature map and determine the confidence level of the keyword by selecting the maximum value between the 14×14 patches and the keyword's text embedding.

A.2 Extended results on NExT-QA and IntentQA

We present extended zero-shot evaluation results on NExT-QA in Table A.12, comparing LVNet with prior zero-shot models across different task categories: causal, temporal, and descriptive reasoning. Models are ordered based on the number of captions processed per video, highlighting the trade-offs between caption efficiency and performance.

LVNet achieves state-of-the-art performance with an overall accuracy of 72.9%, outperforming most models while using only 12 captions per video. Notably, it attains 75.0% on causal reasoning, which is the highest among all models evaluated. For temporal reasoning, LVNet achieves 65.5%, remaining competitive despite using significantly fewer captions than models like VideoTree (56 captions) and LangRepo (90 captions). In descriptive reasoning, LVNet reaches 81.5%, matching VideoTree while processing significantly fewer captions.

Compared to VideoAgent, the closest competing model in terms of caption efficiency (8.4 captions), LVNet demonstrates a substantial performance gain across all categories, with a +2.8% improvement in overall accuracy. While models like VideoTree and TraveLER show strong performance, they process significantly more captions (56 and 65, respectively), indicating that LVNet achieves a superior balance between efficiency and accuracy.

We present extended zero-shot evaluation results on IntentQA in Table A.13, comparing LVNet with prior zero-shot models across different reasoning categories: *Why?*, *How?*, and *B.A.* (Before/After). Models are ordered based on the number of captions processed per video, highlighting the balance between caption efficiency and performance.

LVNet achieves an overall accuracy of 71.7%, outperforming all models while using only 12 captions per video. It achieves 75.0% on the *Why?* category, 74.4% on the *How?* category, and 62.1% on the *B.A.* category. Compared to VideoTree, which processes 56 captions and achieves an overall accuracy of 66.9%, LVNet outperforms it by +4.8% while using significantly fewer captions. Similarly, LangRepo and LLoVi, which process 90 captions, achieve overall scores of 59.1% and 64.0%, respectively, further demonstrating LVNet's caption efficiency.

To ensure fairness, models that utilize video-caption pretraining or process substantially more captions than LVNet are de-emphasized in grey or downplayed in light green in Table A.12 and

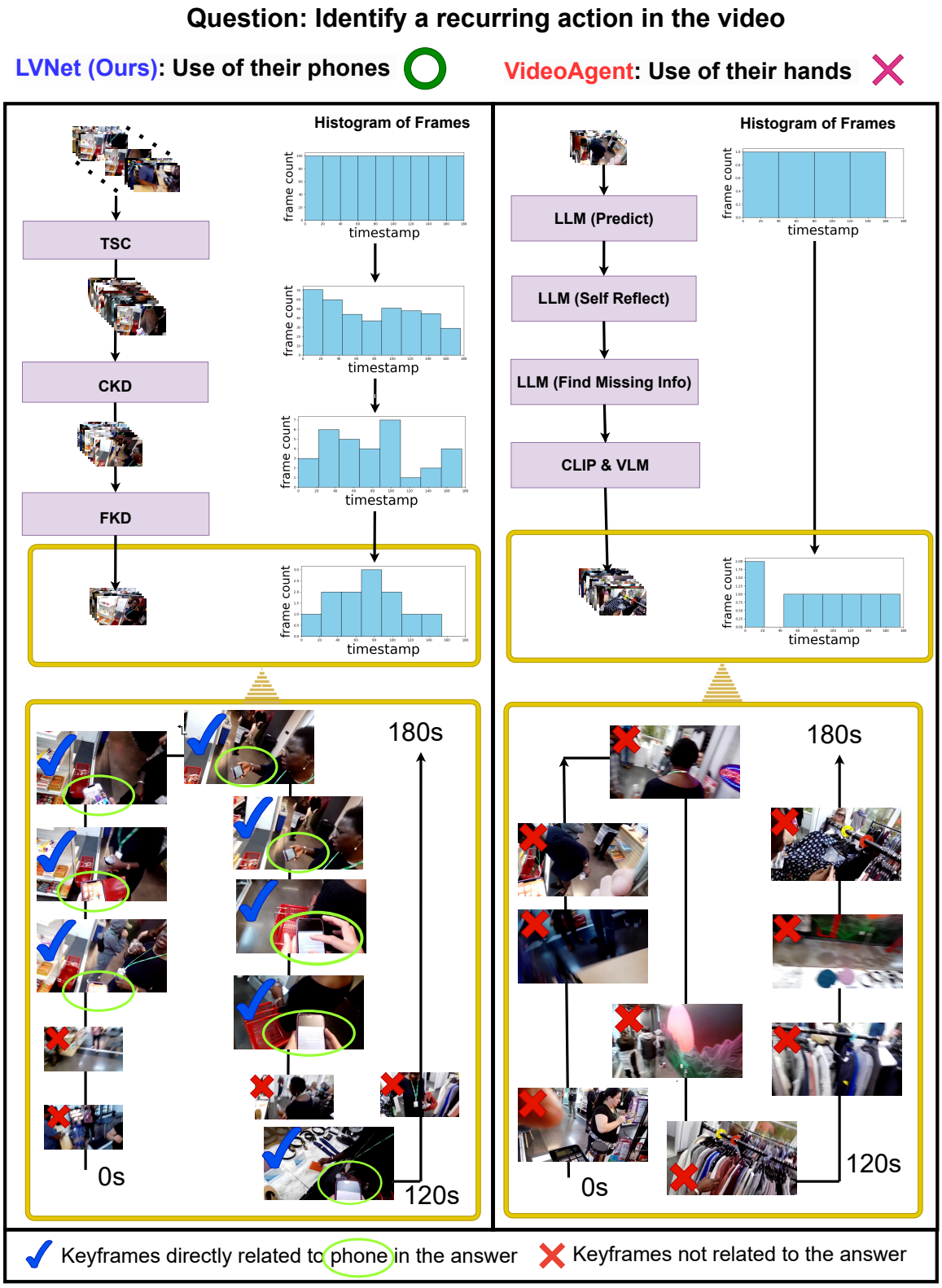

Figure A.4: **Comparison of Keyframe Selection**: Comparison of LVNet and VideoAgent in keyframe selection for video question answering. LVNet refines frames through a multi-stage process (TSC, CKD, FKD) to form a non-uniform keyframe distribution, capturing relevant moments tied to the query. In contrast, VideoAgent relies on uniform sampling and LLM-based frame selection, which fails to focus on crucial keyframes, leading to incorrect predictions. The final keyframe distributions illustrate LVNet's ability to retrieve meaningful frames directly related to the answer, while VideoAgent selects irrelevant frames.

Table A.13.

A.3 Extended results on VideoMME

Table A.14 provides an extended comparison of both *single-stage* and *two-stage* approaches on the VideoMME long-video benchmark. In contrast to single-stage methods, which typically require costly video-level training on large datasets which has the risk of containing evaluation datasets. LVNet is video-level training free and thus retains broader generality. Critically, single-stage pipelines often become infeasible for long videos because they must process entire sequences—potentially exceeding 1,000 frames—through a heavy vision-language model (VLM) or LLM.

Keyframe-selection approaches mitigate this challenge by filtering a minimal set of relevant frames before any large-model processing. As shown in Table A.14, LVNet can handle up to 1,800 frames by reducing them to just a few dozen keyframes, thereby remaining both memory- and compute-efficient while still achieving strong accuracy. For Qwen-VL and GPT4o, we demonstrate an ablation with 1,800 frames to highlight how easily single-stage methods can run out of memory (OOM) under 4 NVIDIA RTX A5000 GPUs or encounter GPT API errors.

Moreover, our approach achieves strong performance with both open-source and proprietary LLM backbones, all without requiring any video-level training. LVNet+DeepSeek-V3 outperforms these single-stage keyframe selection models (VideoChat-T, Frame-Voyager) with equal or less frame selection complexity and outperforms single-stage VideoLLMs with a similarly sized open-source LLM. In a direct two-stage head-to-head setting that uses GPT-4o for both frame captioning and LLM reasoning and enforces the same 24-caption budget, LVNet further surpasses VideoAgent and VideoTree by +2.6% and +1.6%, respectively. Taken together, these results show LVNet's effectiveness over both single-stage and two-stage video methods. The accuracy comes from each model's recommended configuration and matched budgets for fair comparison.

A.4 Qualitative Analysis of Hierarchical Keyframe Selector

We compare open-ended responses of LVNet and the uniform sampling method in Figure A.5 to understand the effectiveness of the hierarchical keyframe selector in LVNet. The frames chosen by LVNet and the naive uniform sampling method are indicated by blue and red checkmarks in the images, respectively. LVNet selects frames at 5, 69, and 135 seconds by executing the hierarchical keyframe selector and generates captions based on those frames. When we feed the concatenated captions to the LLM to answer the given question: *"Based on the video, what are the three main types of tools that C uses..."* in an open-ended manner, the output identifies two main activities: welding torches and measuring tapes, among the three main activities described in Option 3 (welding handle, hammer, measuring tape), which is the correct answer, leading LVNet to choose the correct option.

In contrast, the uniform sampling method selects frames at 0, 16, and 32 seconds and generates captions based on those frames. Similarly, when we feed the concatenated captions to the LLM to answer the same question, the output identifies only one activity—welding tools—resulting in the selection of the incorrect option. This example highlights the importance of keyframe selection and demonstrates the effectiveness of hierarchical keyframe selection in LVNet.

A.5 Algorithms in Detail

Our algorithms are presented in full detail in Algorithm 1, Algorithm 2, and Algorithm 3. TSC in Algorithm 1 extracts per-frame visual features using ResNet-18, followed by an iterative clustering procedure to identify n non-overlapping frame sets. Within each of the n sets, we uniformly sample $\leq \tau$ frames, obtaining a total of $T_a \leq \tau \times n$ frames. For example, LVNet sets $\psi = 5, \lambda = 12, \tau = 18$, resulting in approximately $n \sim 25$ and $T_a \sim 390$ on the EgoSchema dataset. CKD in Algorithm 2 selects top L frames based on similarity/confidence scores, which are calculated using cosine similarity between frames and keywords with CLIP-B/16. LVNet employs $L = 32, len(K) \leq 25$ on the EgoSchema dataset. FKD in Algorithm 3 sorts frames and their corresponding keywords by confidence scores, and reorder the K frames with the lowest scores temporally. It groups frames sequentially into visual templates, each consisting of N frames. From each template, the M frames and keywords most relevant among the N pairs are selected using GPT-4o. We set $L = 32, K = 16, N = 8, M = 3$.

A.6 Prompting: Fine Keyframe Detector

We prompt the VLM to select frames that are most compatible with the list of given keywords. Each

Model	Cap.	Cau. (%)	Tem. (%)	Des. (%)	All (%)
IG-VLM (Kim et al., 2024)	-	69.8	63.6	74.7	68.6
Tarsier (Wang et al., 2024a)	-	-	-	-	79.2
VideoAgent (Wang et al., 2024d)	8.2	72.7	64.5	81.1	71.3
MVU (Ranasinghe et al., 2024)	16	55.4	48.1	64.1	55.2
MoReVQA (Min et al., 2024)	16	70.2	64.6	-	69.2
VFC (Momeni et al., 2023)	32	45.4	51.6	64.1	51.5
SeViLA† (Yu et al., 2024a)	32	61.3	61.5	75.6	63.6
VideoTree (Wang et al., 2024g)	(56)	75.2	67.0	81.3	73.5
ProViQ (Choudhury et al., 2023)	60	-	-	-	64.6
TraveLER (Shang et al., 2024)	(65)	70.0	60.5	78.2	68.2
LangRepo (Kahatapitiya et al., 2024)	90	64.4	51.4	69.1	60.9
LLoVi (Zhang et al., 2023)	90	69.5	61.0	75.6	67.7
LVNet (ours)	12	75.0	65.5	81.5	72.9

Table A.12: **Extended results on NExT-QA (Xiao et al., 2021).** We compare LVNet against prior zero-shot models across different reasoning categories: causal, temporal, and descriptive. LVNet achieves an overall accuracy of 72.9% while using only 12 captions per video, demonstrating strong performance across all reasoning types. Notably, it outperforms all models in causal reasoning (75.0%) and matches the best performance in descriptive reasoning (81.5%), despite processing significantly fewer captions than models like VideoTree (56 captions) and TraveLER (65 captions). Models that utilize video-caption pretraining or process substantially more captions than LVNet are de-emphasized in gray or downplayed in light green to ensure fairness in comparison. Numbers in parentheses () indicate the maximum number of frames used.

Model	Cap.	Why? (%)	How? (%)	B./A. (%)	All (%)
IG-VLM (Kim et al., 2024)	-	-	-	-	65.3
SeViLA† (Yu et al., 2024a)	32	-	-	-	60.9
VideoTree (Wang et al., 2024g)	(56)	-	-	-	66.9
LangRepo (Kahatapitiya et al., 2024)	90	62.8	62.4	47.8	59.1
LLoVi (Zhang et al., 2023)	90	68.4	67.4	51.1	64.0
LVNet (ours)	12	75.0	74.4	62.1	71.7

Table A.13: **Extended results on IntentQA (Li et al., 2023).** We compare LVNet against prior zero-shot models across different reasoning categories: *Why?*, *How?*, and *B.A.* (Belief/Action). LVNet achieves an overall accuracy of 71.7%, surpassing all models while using only 12 captions per video. It reaches 75.0% in the *Why?* category, 74.4% in the *How?* category, and 62.1% in the *B.A.* category. Compared to VideoTree, which processes 56 captions and achieves 66.9% accuracy, LVNet outperforms it by +4.8% while using significantly fewer captions. Additionally, LVNet demonstrates superior reasoning-based performance compared to LangRepo (90 captions, 59.1%) and LLoVi (90 captions, 64.0%). Models with video-caption pretraining or utilizing significantly more captions than 12 frames used by LVNet are de-emphasized in grey or downplayed in light green to ensure fairness with image-level pretraining or highlight caption efficiency. Numbers in parentheses () indicate the maximum number of frames used.

template image contains 8 images, and their order is described in language (e.g. top left to right, bottom left to right) and the VLM outputs the selected images according to our prompting as described in Figure A.6.

A.7 Integration to Existing Methods

Our LVNet has been successfully integrated with other works for evaluation on long video benchmarks. For example, in Ranasinghe et al. (2024) LVNet is integrated with their proposed MVU to gain further performance boosts on the EgoSchema and NextQA benchmarks.

A.8 Comparison with Other Keyframe Selection Methods

We aim to highlight the main advantage of the Hierarchical Keyframe Selector over other existing keyframe selection methods. Models like VideoAgent, VideoTree, and TraveLER provide useful comparisons, as they utilize keyframe selection mechanism with similar or different scale of frames. VideoAgent and TraveLER rely on uniform frame selection in the first iteration without analyzing the entire video even though they perform non-uniform sampling in the next iterations. They identify important segments based solely on these initial frames and the LLM's response, which can be problematic if the initial uniformly selected frames

Method	LLM Active Params/Type	TS	VT Free	# Frames (n)↑	TS Cap. ↓	FR ↑	Complexity	Accuracy ↑
Qwen-VL (Bai et al., 2023)	7B/OS	no	no	4	N/A	–	–	37.8
Qwen-VL	7B/OS	no	no	1800	N/A	–	–	OOM
LongVILA (Chen et al., 2024a)	8B/OS	no	no	128	N/A	–	–	38.8
LongVILA	8B/OS	no	no	256	N/A	–	–	39.7
LLaVA-OneVision (Li et al., 2024)	7B/OS	no	no	8	N/A	–	–	43.8
VideoChat-T (Zeng et al., 2024)	7B/OS	no	no	128	N/A	87.5	$\mathcal{O}(n)$	41.9
Frame-Voyager (Yu et al., 2024b)	7B/OS	no	no	128	N/A	93.7	$\mathcal{O}(n^r)$	48.9
LLaVA-NexT-Video (Zhang et al., 2024a)	34B/OS	no	no	32	N/A	–	–	44.3
Frame-Voyager	34B/OS	no	no	128	N/A	93.7	$\mathcal{O}(n^r)$	51.2
InternVL2 (Chen et al., 2024c)	34B/OS	no	no	16	N/A	–	–	52.6
LVNet (DeepSeek-v3)	37B/OS	yes	yes	1800	24	–	$\mathcal{O}(n)$	53.1
VideoAgent+GPT4o	<1.8T/PP	yes	yes	–	24	–	$\mathcal{O}(m)$	51.3
VideoTree+GPT4o	<1.8T/PP	yes	yes	300	24	92.0	$\mathcal{O}(n^r)$	52.3
GPT4o direct	<1.8T/PP	yes	yes	1800	1800	–	–	API Error
LVNet (GPT-4o)	<1.8T/PP	yes	yes	1800	24	98.7	$\mathcal{O}(n)$	53.9
LLaVA-Video (Zhang et al., 2024b)	72B/OS	no	no	64	N/A	–	–	61.5
Qwen2-VL (Wang et al., 2024c)	72B/OS	no	no	768	N/A	–	–	62.2
InternVL2.5 (Chen et al., 2024b)	72B/OS	no	no	64	N/A	–	–	62.6

Table A.14: **Detailed Comparison on VideoMME (Fu et al., 2024)** . This table expands on Table 3 by including two-stage (TS), video-level training free (VT Free), number of input frames used, two-stage captions fed to LLM (TS Cap.), ratio of frames/captions provided to the LLM relative to input frames (FR), and frame selection complexity. "OS" denotes open-source, "OOM" stands for out-of-memory. $\mathcal{O}(m)$ denotes the complexity proportional to the number LLM calls to predict keyframe timelines. The bottom 3 results are from benchmark leaderboard; we are unable to replicate these on our compute resources. Our LVNet variants require no video-level training yet achieve competitive results on very long videos.

are not representative of the entire video or if the LLM misinterprets the captions and prompts. In such cases, the LLM might incorrectly identify segments for further analysis. If the LLM fails to pinpoint the correct segment initially, the entire process can break down because subsequent frames will be similar to the first set, leading the LLM to continuously select frames within or near the initial segment. Additionally, for videos that are as challenging or more difficult than EgoSchema in terms of temporal complexity and activities, existing keyframe selection models such as VideoAgent, VideoTree, and TraveLER may require numerous iterations by running heavy visual/language models to finalize keyframes selection. This results in higher computational and latency costs, as it necessitates numerous runs of resource-intensive VLM and LLM models.

In contrast, our method analyzes the entire video with high frame rates using a lightweight ResNet-18 (He et al., 2016a) and segments the video non-uniformly based on scene continuity. We then select several frames in each segment by measuring feature similarity between frame features and keywords using the CLIP-B/16 (0.12B) (Ranasinghe et al., 2023) which is lighted than VideoAgent's EVA-CLIP-8Bplus (8B). By reviewing the entire video and non-uniformly selecting keyframes based on scene continuity and similarity scores, these keyframes accurately represent the question-based important frames distribution in the entire video. Furthermore, we use VLM for a fine-grained selection of keyframes, improving keyframe selection when CLIP-B/16 struggles to understand detailed atomic activities in the frames. By hierarchically segmenting the video with different modules, the resulting segments and keyframes are more reliable than those from VideoAgent. Even with more challenging videos, our process only needs to go through the video once to collect keyframes, maintaining computational efficiency.

Figure A.4 visualizes the differences of the keyframe selection mechanism bewtween LVNet and VideoAgent. On the left, LVNet begins with uniformly sampled frames and filters them through multiple stages, resulting in a non-uniform distribution of frames over time. First, the temporal scene clustering (TSC) selects some frames that represent temporally distinct activities. Next, the coarse keyframe detector (CKD) targets frames most relevant to the question. Finally, the fine keyframe detector (FKD) further refines this selection to ensure the keyframes accurately capture the activity in question. As a result, LVNet produces 12 frames, with 8 of them (67%) directly depicting "usage of phones," which is the correct answer and leads the model to select the right option. On the right, VideoAgent also starts with the uniform frames but relies on a LLM to request additional frames. Since the initial frames do not capture enough relevant content, the LLM again selects frames uniformly, adding more irrelevant samples that lack the crucial information about "usage of phones." As a result, VideoAgent ultimately selects

Time(s):	0	5	16	32	69	135
Caption:	A person is using a hacksaw…	… holding a welding torch …	A person is welding or soldering …	C is using a power tool …	A person is working on unwrapping…	…They are manipulating a measuring tape …

Prompt: {Concatenated Captions from ✓ or ✗ }. I request you to answer the following question based on the preceding descriptions in less than 50 words. Question: Based on the video, what are the three main types of tools that c uses, and how do their roles in shaping the iron differ from one another?

LLM Output:

✓ **LVNet (Ours)**: Based on the video, the three main types of tools used are welding torches, pliers, and measuring tapes…

✗ **Uniform Sampling**: The three main types of tools used are hacksaws, welding tools, and power tools…

GT (option 3): The three main types of tools that c uses are a welding handle, a hammer, and a measuring tape…

Figure A.5: **Open-ended Responses from LVNet vs Uniform Sampling**: The frames chosen by LVNet and the naive uniform sampling method are indicated with blue and red checkmarks, respectively. LVNet identifies both welding torches and measuring tapes, choosing the correct option, whereas uniform sampling only detects welding tools and selects the incorrect answer. The blue, red, and purple highlights correspond to the three main activities in the video—welding a handle, using a hammer, and using a measuring tape, respectively.

the wrong option.

Algorithm 1: Temporal Scene Clustering

Require: ResNet-18 (He et al., 2016b) pretrained on imagenet dataset f, frame list $\textbf{List}_{frame}$, image index list $\textbf{List}_{index} \in \{1, \ldots, N\}$, minimum number of list length ψ, temperature λ, number of sample τ, function to find index of x in list $\mathbf{w}$ **index**(x, $\mathbf{w}$), and function to sort list **sort(List)**

for all img^i `in` $\textbf{List}_{frame}$ **do**
  $\mathbf{F}^i \leftarrow f(img^i)$
  $\textbf{List}_{feat}$`.insert(`$\mathbf{F}^i$`)`
end for
for all $\mathbf{F}^i$ `in` $\textbf{List}_{feat}$ **do**
  $\textbf{List}_{dist} \leftarrow \frac{\sum_y \sum_x \sqrt{(\mathbf{F_i} - \textbf{List}_{feat})^2}}{x \times y}$
  $\mathbf{M}_{dist}$`.insert(`$\textbf{List}_{dist}$`)`
end for
while `length of` $\textbf{List}_{index} > \psi$ **do**
  $\textbf{List}_{sample} \leftarrow \emptyset$
  $\textbf{List}_{\delta} \leftarrow \emptyset$
  $i \leftarrow \textbf{List}_{index}$`.pop(0)`
  $\mathbf{p}^i \leftarrow$ `softmax(`$\mathbf{M}^i_{dist}$`)`
  $\mu_{\mathbf{p^i}}, \sigma_{\mathbf{p^i}} \leftarrow$ `mean(`$\mathbf{p}^i$`), std(`$\mathbf{p}^i$`)`
  $\beta \leftarrow \mu_{\mathbf{p^i}} - \sigma_{\mathbf{p^i}} \sum_{i=0} e^{1-i/\lambda}$
  for all `prob in` $\mathbf{p}^i$ **do**
    if `prob` $< \beta$ **then**
      $\textbf{List}_{selected}$.insert(**index**(`prob`, $\mathbf{p}^i$))
    end if
  end for
  for all γ in $\textbf{List}_{selected}$ **do**
    $\delta \leftarrow \gamma\ th$ `value in` $\textbf{List}_{index}$
    $\textbf{List}_{\delta}$`.insert(`δ`)`
    $\textbf{List}_{index}$`.pop(`γ`)`
  end for
  $\textbf{List}_{\delta}$`.insert(`i`)`
  $\textbf{List}_{sample} \leftarrow$ `sample` τ `items from` $\textbf{List}_{\delta}$
  sort($\textbf{List}_{sample}$)
  for all $frame^j$ in $\textbf{List}_{frame}$ **do**
    if j `in` $\textbf{List}_{sample}$ **then**
      Outputs`.insert(`$frame^j$`)`
    end if
  end for
end while

Algorithm 2: Keyword-Image Matching Process in CKD

Require: keyword set $\mathbf{K}$, image set $\mathbf{I}$, total length of selected image set L, function to calculate similarity matrix **sim**($\mathbf{K}, \mathbf{I}$), function to sort similarity matrix and return indices **sort**($\mathbf{S}$)

$\mathbf{S} \leftarrow$ **sim**($\mathbf{K}, \mathbf{I}$)
$\mathbf{S}_{sorted}, \mathbf{idx}_{sorted} \leftarrow$ **sort**($\mathbf{S}$)
Initialize $\mathbf{P}_{best}$ as an empty list
Initialize $\mathbf{I}_{selected}$ as an empty set
while `length of` $\mathbf{I}_{selected} < L$ **do**
  for $k \in \mathbf{K}$ **do**
    for $i \in \mathbf{I}$ **do**
      $i_{index} \leftarrow \mathbf{idx}_{sorted}[k][i]$
      if i_{index} `not in` $\mathbf{I}_{selected}$ **then**
        $\mathbf{P}_{best}$.insert(k, i_{index})
        $\mathbf{I}_{selected}$.insert(i_{index})
        break
      end if
    end for
    if `length of` $\mathbf{I}_{selected} \geq L$ **then**
      break
    end if
  end for
end while
return $\mathbf{P}_{best}$

Algorithm 3: Fine Keyframe Detection Process (FKD)

Require: keyword set $\mathbf{K}$, image set $\mathbf{I}$, similarity score list $\mathbf{S}$, total length L, number of low similarity indices K, number of frames per visual template N, number of keyframes selected per visual template M, function to sort by similarity **sort**($\mathbf{S}$), function to order indices temporally **temporal_order**()

$\mathbf{idx}_{sorted} \leftarrow$ **sort**($\mathbf{S}$)
$\mathbf{idx}_{low\_sim} \leftarrow \mathbf{idx}_{sorted}[-K:]$
$\mathbf{idx}_{temporal} \leftarrow$ **temporal_order**($\mathbf{idx}_{low\_sim}$)
$\mathbf{idx}_{final} \leftarrow$ `concatenate(`$\mathbf{idx}_{sorted}[:-K], \mathbf{idx}_{temporal}$`)`
$\mathbf{I}_{ordered}, \mathbf{K}_{ordered} \leftarrow \mathbf{I}[\mathbf{idx}_{final}], \mathbf{K}[\mathbf{idx}_{final}]$
sets $\leftarrow$ `create_sets(`$\mathbf{I}_{ordered}, \mathbf{K}_{ordered}, L//N$`)`
for each set $\in$ **sets do**
  $\mathbf{I}_{selected} \leftarrow$ `select_top_M(`**set**$, M$`)`
end for
return $\mathbf{I}_{selected}$

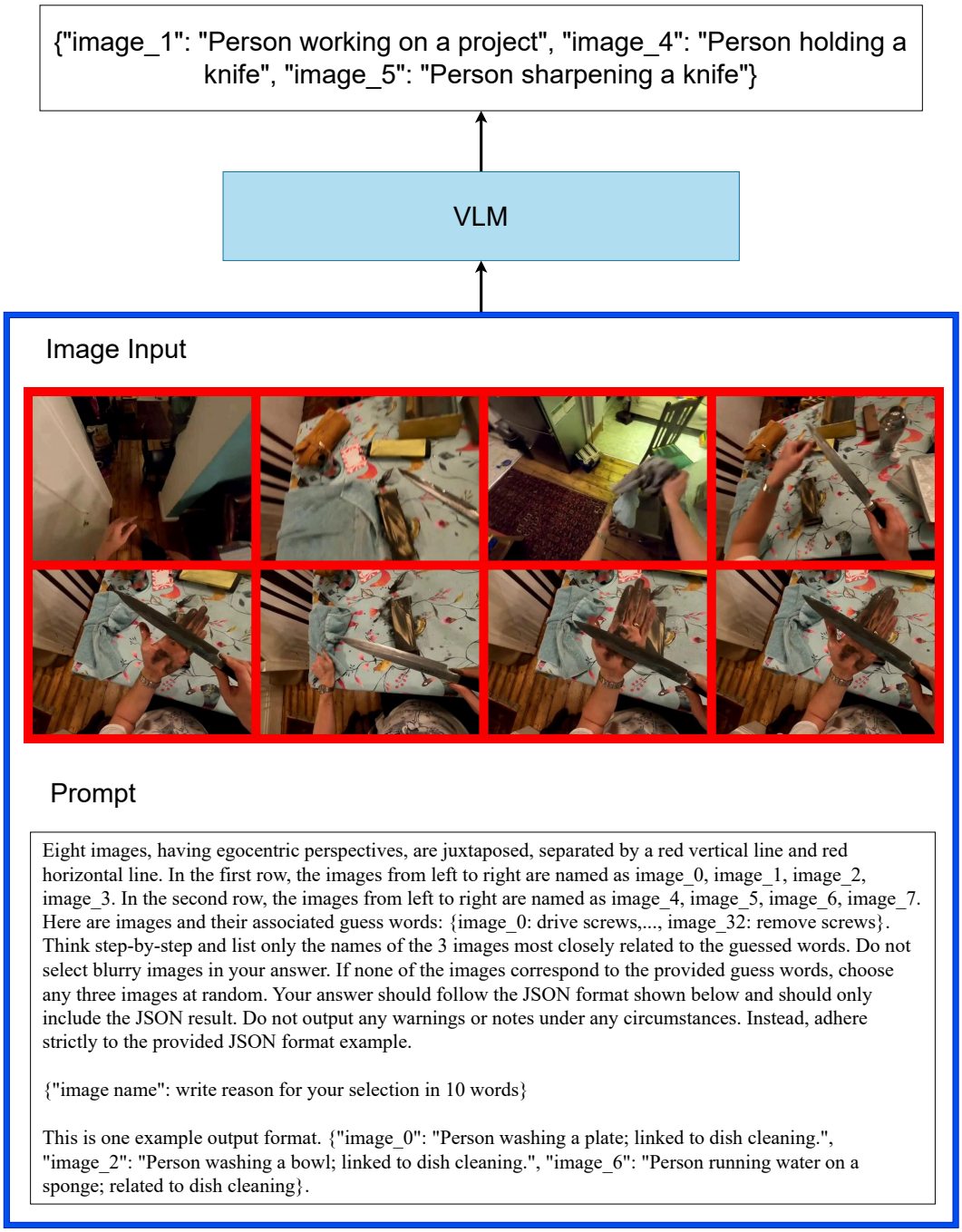

Figure A.6: **Prompt for Fine Keyframe Detection**: The figure illustrates the input image, the prompt provided to the VLM, and the output. The input image represents a visual template composed of eight frames, and the prompt requests the three best frames along with their corresponding keywords. The output displays the top three selected frames and their associated keywords.